Universitetet i Oslo
Institutt for informatikk

Estimering av en straffet Cox-modell på bakgrunn av mikroarray genekspresjonsdata

Hege Leite Størvold
hegelst@ifi.uio.no

26. oktober 2004

Forord

Denne oppgaven er en del av Cand. Scient-graden i informatikk, studieretning databehandling, ved Institutt for Informatikk ved Universitetet i Oslo. Arbeidet har blitt gjennomført fra høsten 2002 til høsten 2004 ved Institutt for Informatikk, UiO. Jeg vil rette en spesiell takk til min veileder Ole Christian Lingjærde for gode råd og tålmodighet underveis. Jeg vil i tillegg takke Ørnulf Borgan, Johan Fosen og Ståle Nygård for god hjelp med oppgaven.

Oppgaven er skrevet på norsk og det har blitt lagt vekt på å bruke norske faguttrykk så sant dette har vært mulig. Oppgaven er skrevet med tanke på lesere som har en viss bakgrunn i statistikk og statistisk analyse.

Innhold

Kapittel 1

Problemstilling

1.1 Innledning

En sentral problemstilling innen medisinsk statistikk er å finne faktorer som gir informasjon om overlevelsestiden til individer. Begrepet overlevelsestid brukes her i vid forstand og kan benevne enten levetiden til et individ, eller tiden til en annen definert hendelse i individets livsløp. Et eksempel av særlig relevans for denne oppgaven, er at individet har en bestemt sykdom (for eksempel kreft) og at man ønsker å forklare resterende levetid eller tid til tilbakefall etter behandling, ut fra genetiske faktorer. Slike og liknende analyser er innen statistikken kjent som *overlevelsesanalyse* og er i ferd med å bli et viktig felt innenfor bioinformatikk. Utgangspunktet for overlevelsesanalyse innen bioinformatikk er typisk at man for et antall individer har observert aktivitet (ekspresjon) til tusenvis av gener og overlevelsestid. Aktiviteten til et gen - det vil si raten som informasjonen i genet leses av og omsettes til et genprodukt - avhenger av en rekke faktorer, inkludert celletype og celletilstand (blant annet om cellen er normal eller avviker på noen måte, slik man for eksempel ser ved kreft). Målsetningen ved å utføre analysen er å finne hvilke geners ekspresjon som best gir informasjon om overlevelse.

Dette kapittelet starter med en kort innføring i genuttrykk og mikroarrayer, og hvilke muligheter som finnes for bruk av dette innenfor overlevelsesanalyse. Videre følger en introduksjon til statistisk analyse av denne typen data, hvilke problemstillinger som kan være interessante og hvilke utfordringer som ligger i å analysere slike datasett. Kapittelet avsluttes med en presentasjon av den konkrete målsetningen for denne oppgaven og en kort oversikt over strukturen på oppgaven.

1.2 Genekspresjon og mikroarrayer

Så nær som alle celler i mennesket inneholder det samme genmaterialet, som er oppskriften på hvordan alle cellene skal se ut og fungere. Kroppen er satt sammen av en rekke (ca. 200) ulike typer celler, og det som skiller de ulike celletypene fra hverandre er blant annet hvilke gener som er aktivert til enhver tid. Det at et gen er aktivt, eller uttrykkes, vil si at det oversettes fra

DNA, der selve informasjonen er lagret, til et funksjonelt protein eller et annet genprodukt. Det er hvilke proteiner som lages i cellen som bestemmer mesteparten av dens utseende og funksjon. Mekanismene som regulerer hvilke gener som er aktive i en celle er nøyaktige og sensitive, og de gjør det mulig for cellen å dynamisk tilpasse seg indre og ytre endringer. Hvis man for eksempel blir syk vil ulike celler i kroppen kunne endre produksjonsmengden av en type protein, eller produsere andre proteiner enn det som er normalt i cellen ved frisk tilstand.

I 1990 ble det Humane Genomprosjektet (HGP) startet. Prosjektet hadde som mål å kartlegge det menneskelige genomet. Dette innebar blant annet å bestemme sekvensen til de omlag $3,2 * 10^9$ baseparene i den menneskelige genomet, i tillegg til å identifisere alle genene. Kartleggingen av selve sekvensen ble ferdig i 2003. Dette vil si at man nå med høy presisjon kjenner rekkefølgen til baseparene. Man mangler imidlertid en full forståelse av denne sekvensen. Man vet at det finnes omlag $30.000 - 35.000$ gener, men vet ennå ikke hva alle genene koder for. Fremdeles er funksjonen til 50% av de genene man allerede har funnet ukjent. En utfordring videre er derfor å kunne forstå hvilken funksjon hvert gen har, og hvordan de ulike genene virker sammen. Et interessant mål er å plukke ut hvilke gener som endrer aktivitet ved utbrudd eller utvikling av sykdom ved å se på mønstre i genaktivitet som korrelerer med sykdom eller død. Man kan for eksempel studere hvilke endringer i genaktivitet som oppstår i cellene i en tumor i forhold til tilsvarende friske celler. Dersom man kan finne hvilke genekspresjonsmønstre som kjennetegner ulike sykdommer er det nærliggende å tro at det er mulig å bruke dette til å bedre forståelsen av sykdommen. Man kan ved hjelp av dette for eksempel stille mer presise diagnoser og finne bedre behandlinger. I tillegg kan man håpe å finne faktorer som predisponerer for ulike sykdommer, og få en bedret forståelse av de ulike biologiske mekanismene som spiller inn i de forskjellige stadiene av et sykdomsforløp.

Mikroarrayer gjør det mulig å måle aktiviteten til tusenvis av gener samtidig både enkelt, raskt og effektivt. Det at et gen uttrykkes vil si at det kopieres fra DNA til et mellomprodukt, mRNA, som igjen oversettes til et bestemt protein. Man måler uttrykket av alle gener i en celle ved å isolere alt mRNA i cellen. Mengden av mRNA-molekyler for et bestemt gen brukes som mål på hvor mye dette genet er uttrykt i cellen. Ulike celler har ulikt genuttrykk, og det er denne forskjellen man som regel er interessert i å måle. Genuttrykk kan skille friskt fra sykt vev, påvise behandlingseffekt og gi informasjon om enkeltgener eller samspillet mellom gener. Mikroarrayer har mange bruksområder innenfor medisin og biologi[46]. Kapittel 2 tar for seg genaktivitet og detaljene rundt hvordan mikroarrayer kan brukes til å måle genaktivitet.

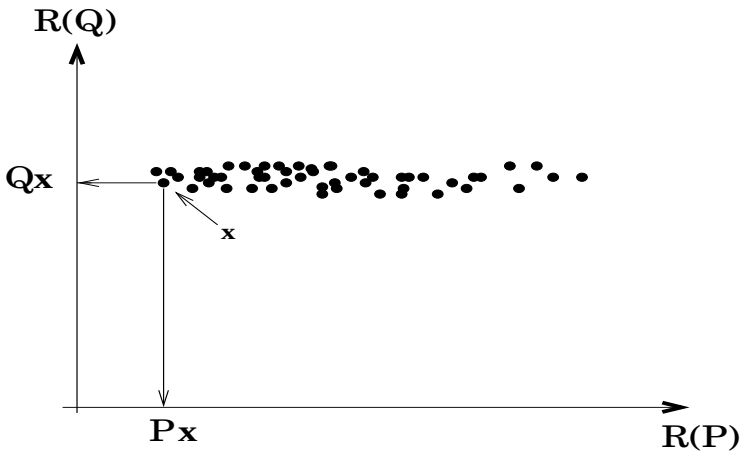
Det er ønskelig å komme fram til metoder som gjør det mulig å finne alle genene som gir informasjon om sykdom eller overlevelse. I og med at man i utgangspunktet ikke vet hvilke gener dette er, måles ekspresjonen til så mange gener som mulig i et mikroarrayforsøk. Data fra mikroarrayforsøk kan derfor inneholde noen gener med ekspresjon som er relevant med hensyn på overlevelse, men også mange gener som ikke er det. Det beste ville være om man kunne måle uttrykket til alle genene i en celle, for ikke å overse potensielt relevante gener. Man kan studere både kjente og ukjente gener ved å ekstrahere informasjonen om alle aktive gener i en celleprøve (se på total-mRNA). Ved å studere ekspresjonen til alle gener kan man blant annet prøve å lære noe om funksjonen til genene. Det å kunne plukke ut gener som er relevante for overlevelse kompliseres av at en del gener kan være knyttet til samme reaksjonsvei (pathway), det vil si at de koder for proteiner som inngår i de samme biologiske prosessene. Slike gener kan derfor noen ganger være likt regulert, ofte kalt *samregulerte*, og kan ha rimelig likt ekspresjonsmønster. Dette resulterer ofte i høy grad av korrelasjon mel-

lom ekspresjonsmønstrene i de målte ekspresjonsdataene, noe som i mange tilfeller fører til at modelleringen av slike data kan bli forholdsvis komplisert. Biologene er ofte interessert i å forstå slike sammenhenger og å kunne identifisere hvilke biologiske mekanismer som spiller inn under ulike sykdommer. Det er en utfordring å kunne identifisere samregulerte gener og å kunne plukke ut de genene som er mest relevante, for eksempel for overlevelse. For å kunne si noe om hvilke gener som er relevante kan man gjøre en statistisk analyse av dataene.

1.3 Statistisk analyse av mikroarraydata

Forsøk med mikroarrayer produserer store mengder data. En utfordring er å kunne gjøre en statistisk analyse av slike data for å kunne hente ut informasjonen som ligger i dem. Målet med å gjøre en statistisk analyse av overlevelsesdata kan for eksempel være å bruke dataene til å tilpasse en passende parametrisk modell. Hensikten med å tilpasse en slik modell er å finne sammenhengen mellom en rekke inputvariable (kovariater) og en responsvariabel. I denne oppgaven vil kovariatene være målte genekspresjonsverdier for p ulike gener, mens responsen vil være en levetid t. En slik parametrisk modelltilpasning (eller *regresjon*) resulterer i en parametervektor $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$, der størrelsen på koeffisienten β_i uttrykker hvor viktig det i'te gen er for å forklare overlevelsen. I denne oppgaven vil definisjonen av et gen som har betydning for, eller er relevant for overlevelsen være at en endring i ekspresjonen av dette genet har en relevant prediksjonsverdi i en statistisk modell estimert på bakgrunn av et observert datasett. Tilsvarende vil definisjonen av et gen uten betydning for levetiden være at en eventuell endring i ekspresjonen av dette genet ikke har noen prediksjonsverdi i en modell estimert på bakgrunn av disse dataene. Det vil altså ikke menes at disse genene nødvendigvis har noen større eller mindre biologisk effekt på hvor lenge et individ lever i forhold til andre gener. Videre studier av genene vil eventuelt kunne gi oss innsikt i dette.

Et karakteristisk trekk ved data fra mikroarrayforsøk som kompliserer analyser av denne typen data, er at man ofte har et forholdsvis lite antall individer n, der n sjelden er over et par hundre. For hvert individ måles ekspresjonen til et stort antall gener, p, ofte opptil noen titusener. Antall målte variable er derfor som regel mye større enn antall individer og denne høye dimensjonaliteten i dataene gjør at det blir problematisk å bruke standard statistiske analysemetoder. Dette henger sammen med at hvor godt man klarer å modellere en sammenheng mellom kovariatene og responsen blant annet avhenger av antall kovariater som inkluderes i modellen. Dersom antallet parametere som skal estimeres øker, dvs. antallet ukjente vokser, vil det intuitivt sett være nærliggende å tro at man må øke datamengden for å kunne estimere alle de ukjente parameterne med en viss sikkerhet. Teoretisk sett kan det vises at dette stemmer. Antall observasjoner som trengs for å kunne estimere $\boldsymbol{\beta}$ med en viss nøyaktighet vokser i mange tilfeller eksponentielt med antall kovariater, det vil si antall gener som er med i regresjonen i dette tilfellet. Dette fenomenet kalles ofte *dimensjonalitetens forbannelse*[12]. Den fundamentale årsaken til fenomenet er at funksjoner av høy dimensjon har potensiale til å være mye mer kompliserte enn lav-dimensjonale funksjoner, og slike komplikasjoner er vanskelige å unngå. I et mikroarrayforsøk er det ikke praktisk mulig å øke antall observasjoner betydelig, ettersom det begrenses av antall pasienter man har med i undersøkelsen. I tillegg vil antall gener alltid være stort, ettersom dette er en nødvendighet når man ønsker å finne hvilke gener som er relevante for en gitt respons. I denne sammenhengen er det derfor ikke praktisk mulig å unngå dette høydimensjonale problemet ved å øke antall observasjoner.



Figur 1.1: Illustrasjon av data som projiseres ned i to underrom. For disse dataene vil projeksjonen $P\mathbf{x}$ fange opp det meste av spredningen i data, mens projeksjonen $Q\mathbf{x}$ fanger opp lite av spredningen, og blir tilnærmet konstant.

En egenskap man ofte finner i høydimensjonale data er at mesteparten av spredningen i dataene er knyttet til et underrom av lav dimensjon. I de øvrige retningene av rommet har dataene lav spredning, og følgelig vil en funksjon av dataene være tilnærmet konstant i slike retninger. Dette kan illustreres grafisk med et lite eksempel. Anta at vi har data $\mathbf{x} \in \mathbb{R}^p$ og en funksjon $f(\mathbf{x})$ for slike data, slik at $f : \mathbb{R}^p \to \mathbb{R}$. La f være en lineær funksjon, det vil si at $f(\alpha\mathbf{x} + \beta\mathbf{y}) = \alpha f(\mathbf{x}) + \beta f(\mathbf{y})$ for alle $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ og $\alpha, \beta \in \mathbb{R}$. Dersom P er en projeksjonsmatrise, så vil $P\mathbf{x}$ være en projeksjon av dataene $\mathbf{x}$ ned i underrommet utspent av kolonnene i P, og $\mathbf{x}$ kan følgelig skrives som summen av projeksjonen ned i dette underrommet og projeksjonen ned i det ortogonale underrommet utspent av kolonnene i $Q = (I - P)$. Figur 1.1 viser grafisk hvordan slike projeksjoner kan se ut. Dersom dataene $\mathbf{x}$ har minimalt med spredning i underrommet utspent av kolonnene i Q, det vil si $Q\mathbf{x} \approx \boldsymbol{c}$ for alle $\mathbf{x}$ i datasettet, hvor $\boldsymbol{c} \in \mathbb{R}^p$ er en konstant, så vil

$$f(\mathbf{x}) = f(P\mathbf{x}) + f(Q\mathbf{x}) \approx f(P\mathbf{x}) + f(\boldsymbol{c})$$

For det gitte datasettet vil altså f være tilnærmet konstant i underrommet Q.

Denne egenskapen ved høydimensjonale data kan utnyttes for å unngå dimensjonalitetens forbannelse, ved å bruke såkalte regulariseringsmetoder. Dette er metoder som på ulike måter vektlegger de dimensjonene i dataene som bidrar mest til responsen, samtidig som dimensjoner med lite varians vektlegges mindre. Ved å regularisere håper man å få en mer stabil funksjon som gir mindre varians i estimatene av $\boldsymbol{\beta}$ under en modelltilpasning. Regulariseringsmetoder er et viktig verktøy i en høydimensjonal situasjon. Mange av metodene har vært i bruk lenge og det har blitt opparbeidet en god statistisk forståelse av dem. Ridge regresjon er en av de enkleste og mest brukte regulariseringsteknikkene. Denne typen regularisering har i praksis vist seg å være et nyttig verktøy i mange situasjoner der man ønsker å analysere høydimensjonale data. Blant annet innen fagfelt som innebærer bildeanalyse har man hatt stor suksess ved å bruke ridge regresjon og liknende regulariseringsteknikker. Et eksempel som kan nevnes her er såkalt computer tomografi[16]. Dette går ut på å danne seg et bilde av for eksempel en svulst inne i et legeme ved å sende stråler gjennom legemet og måle hvor mye stråling

som absorberes i legemet. Det å hente ut informasjon om formen på selve svulsten blir i slike tilfeller et ekstremt ill-posed problem (se kapittel 4 for forklaring), ettersom data angående selve svulsten forstyrres av data angående legemet rundt svulsten. I slike tilfeller kan man likevel få et veldig godt bilde av svulsten basert på denne mangelfulle informasjonen, ved å bruke ulike regulariseringsteknikker.

Ettersom ridge regresjon har vist seg å være verdifull i slike liknende høydimensjonale situasjoner, ble det derfor ansett som interessant å teste ut en ridge-type regularisering i denne oppgaven. Metoder for analyse av overlevelsesdata vil bli gjennomgått i kapittel 3 og metoder for regularisering, med spesiell vekt på ridge regresjon, vil bli gjennomgått i kapittel 4.

1.4 Mål med oppgaven

I kapittel 4 vil det bli presentert ulike tilnærminger som tidligere har vært foreslått for å knytte sensurerte overlevelsesdata opp mot ekspresjonsdata, både med den hensikt å kunne plukke ut gener med ekspresjonsmønstre som er assosiert med overlevelsestider og for å kunne bygge statistiske modeller som kan brukes til prediksjon av overlevelse. Det vil i denne oppgaven bli bygget videre på deler av dette arbeidet. Oppgaven tar for seg en av de vanligste statistiske modellene for overlevelsesdata, den såkalte *Cox proporsjonale hasardmodell* (eller bare *Cox-modellen*). Denne brukes vanligvis på datasett med langt flere observasjoner (individer) enn kovariater. Vi skal se hvordan denne modellen kan tilpasses til en situasjon der antall kovariater er langt høyere enn antall observasjoner, slik situasjonen er når de observerte kovariatene består av målinger av ekspresjonen til tusenvis av gener. Generelt er det en rekke problemstillinger knyttet til bruk av Cox-modellen. Relevante spørsmål kan være:

- Hvordan kan vi oppnå best mulig prediksjon av overlevelse for nye individer?
- Hvordan kan vi oppnå best mulige estimater for regresjonsvektoren $\boldsymbol{\beta}$?
- Hvordan kan vi estimere usikkerheten til de estimerte parameterne og til levetidsprediksjonene?
- Hvilke gener (kovariater) er viktigst å ha med i modellen?

I denne oppgaven vil et av hovedmålene være å estimere en regularisert Cox-modell på bakgrunn av alle genene fra et mikroarraydatasett. Regulariseringen som benyttes i denne oppgaven vil være en L_2-type straff. En straffet modell kan blant annet brukes til prediksjon av overlevelse for nye individer. Ved prediksjon av overlevelse kan den estimerte modellen brukes som en slags sort boks som predikerer overlevelsen til et individ, eller eventuelt plasserer det i en risikogruppe. Under visse forutsetninger vil en regularisert modell kunne gi lavere prediksjonsfeil enn en uregularisert modell, i og med at man ved å bruke regularisering fjerner noe av støyen i dataene.

En annen problemstilling det vil bli fokusert på er hvordan en variabelseleksjon basert på dataene kan utføres, det vil si hvordan plukke ut et antall gener som er relevante for overlevelsen. En enkel tilnærming til å gjøre dette er å se på hvert gen i den tilpassede modellen

isolert og si om dette genet er relevant i den gitte sammenheng eller ikke. En mer kompleks tilnærming vil kunne være å analysere grupper av gener samtidig og se om disse genene tilsammen har relevans. Slike relevante grupper av gener vil kunne gi ideer om regulatoriske reaksjonsveier som er involvert i situasjonen som studeres. I denne oppgaven vil det kun bli fokusert på den enkleste tilnærmingen der relevansen av ett og ett gen vurderes for seg. I og med at dataene har en iboende biologisk kompleksitet, som nevnt i avsnitt 1.2, er det ikke sikkert at det vil være mulig å plukke ut gener som har betydning for overlevelsen. Dersom det finnes flere gener som har ekspresjonsmønstre som likner, det vil si er høyt korrelerte, vil metoden kunne få problemer med å skille ut det eller de genene som faktisk er relevante. Selv med en regularisert estimert modell er det ikke sikkert at dette vil gi gode resultater når det kommer til variabelseleksjon. Variansanalyse vil trolig også være noe komplisert ved høy dimensjonalitet i data og med bruk av regulariserte metoder, dersom det er ønskelig å teste for signifikans. Ulike tilnærminger til genseleksjon vil bli tatt opp i kapittel 7.

Det finnes ulike metoder for å kunne teste hvilken regularisert modell som er best etter ulike kriterier. Et av de primære målene med denne oppgaven vil være å teste ulike modellseleksjonsmetoder for regulariserte Cox-modeller som har blitt estimert på bakgrunn av mikroarraydata. Disse modellseleksjonsmetodene vil i hovedsak være kryssvalidering og L-kurvekriteriet.

Arbeidet vil omfatte følgende steg:

1. Simulere genekspresjons- og overlevelsesdata fra en Weibull akselerert levetidsmodell til bruk under studien.
2. Velge estimeringsmetode for Cox-modellen. Dette vil innebære å evaluere og implementere algoritmer for tilpasning av Cox-modellen.
3. Evaluere hvordan tilpasning av modellen påvirkes av ulike parametere, som for eksempel antall individer, antall gener og antall gener av betydning, når vi antar at vi har en fast straffeparameter. Evalueringen vil bli gjort ved å se på prediksjonsfeil og ulike kriterier for genseleksjon.
4. Implementere og evaluere modellseleksjonsmetoder for å estimere en optimal straffeparameter. Evalueringen vil igjen gå ut på å sammenlikne prediksjonsfeil og hvor godt det er mulig å gjøre en genseleksjon på bakgrunn av modellen.

1.5 Oversikt over oppgaven

I kapittel 2 vil den biologiske bakgrunnen for mikroarrayer og prinsippene ved mikroarrayteknologien bli presentert. I kapittel 3 gir en oversikt over den statistiske bakgrunnen for overlevelsesanalyse og Cox-modellen. Regulariseringsteknikker, med vekt på ridge regresjon vil bli presentert i kapittel 4, og optimeringsmetoder for å kunne estimere en straffet Cox-modell vil bli gjennomgått i kapittel 5. Detajer rundt ulike metoder for modellseleksjon for en straffet Cox-modell vil bli tatt for seg i kapittel 6. Metoder for genseleksjon på bakgrunn av en straffet Cox-modell vil bli omtalt i kapittel 7. Resultater fra tester av metodene vil bli presentert i kapittel 8, mens kapittel 9 inneholder en diskusjon rundt disse resultatene samt forslag til videre arbeid.

Kapittel 2

Mikroarrayer

Et menneske består av en stor mengde celler (ca. 75.000.000.000.000). Felles for alle disse cellene er at de har hver sin kopi av vårt arvemateriale, DNA, som inneholder informasjon om hvordan hele vår organisme er bygget opp. Vårt arvemateriale består av anslagsvis 30.000-35.000 gener, eller kodende enheter[49]. Forskjellig *uttrykk* av disse genene i ulike celler gir opphav til forskjellige celletyper. Genuttrykk eller genekspresjon er prosessen der gener leses av og oversettes til proteiner. Uttrykket av gener varierer med vevstype, stadier av organismens utvikling, ved sykdom og ved påvirkning fra omgivelsene, både på cellulært nivå og på organismenivå. Hvilke gener som uttrykkes i en celle til enhver tid er styrt av en rekke mekanismer som påvirkes av faktorer både inni og utenfor cellen. Selv om man i dag kjenner til en del av mekanismene som benyttes for å regulere genuttrykket i en celle er det svært langt igjen før man forstår i detalj hvordan disse virker sammen og hva som regulerer dem. Et relativt nytt og viktig verktøy for å kunne måle uttrykket til titusener av gener på en gang er mikroarrayteknologien. Ved hjelp av denne teknologien er man i stand til å kartlegge en celles globale genuttrykk, og man kan måle en celles globale respons på forandringer i dens miljø, eller interne forandringer på grunn av endringer i dens genmateriale. Resultater fra en slik analyse av genuttrykk kan bidra til forståelsen av geners funksjon og forståelsen av genregulering og interaksjoner i en organisme.

Det første avsnittet i kapittelet gir en enkel introduksjon til den biologiske bakgrunnen for mikroarrayteknologien. Videre vil mikroarrayer og bruken av disse bli presentert nærmere.

2.1 Biologisk bakgrunn

Enhver levende organisme består av en eller flere celler. En kompleks organisme, som for eksempel mennesket, kan bestå av milliarder av celler fordelt på ulike typer. Alle cellene i kroppen, med unntak av røde blodlegemer, inneholder det samme arvematerialet. Dette arvematerialet som finnes i hver celle kalles et *genom*. Genomet består av flere *kromosomer* som hvert inneholder et stort DNA-molekyl. Til sammen inneholder kromosomene all informasjon som er nødvendig for at en organisme skal utvikles, vedlikeholdes og reprodusere seg. Gener spiller en sentral rolle i denne sammenheng. Hvert gen er et segment av et DNA-molekyl og

har en funksjon, vanligvis å spesifisere aminosyresekvensen til et protein. Vi skal først se litt nærmere på DNA, før vi kommer tilbake til gener og genaktivitet.

2.1.1 DNA

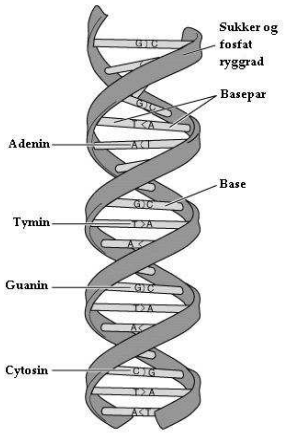
DNA eller *deoxyribonukleinsyre* er den genetiske informasjonsbæreren. DNA-molekylet er bygd opp av fire ulike byggestener, kalt nukleotider. En nukleotide består av et sukkermolekyl (deoxyribose), et fosfatmolekyl og en av de fire basene adenin(A), tymin(T), cytosin(C) og guanin(G). Nukleotidene bindes sammen etter hverandre i en kjede, og dette kalles en polynukleotidkjede. Hver DNA-kjede har en retning. I den ene enden av en slik kjede vil det være deoxyribose med en fri 3'-OH (*3'-enden*) og i den andre enden vil det være deoxyribose med en fri 5'-fosfat (*5'-enden*). Når man lister rekkefølgen til basene i et DNA-molekyl leses de alltid i *positiv retning*, definert til å være fra 5'-enden til 3'-enden. Den vanligste formen til DNA i celler er at to slike nukleotidkjeder er bundet sammen ved at to og to av basene i hver kjede pares, slik at det dannes såkalt *dobbeltrådet DNA*[46]. Reglene for sammenbinding av basene gjør at A kun kan bindes med T og C kun kan bindes til G. Man sier at A og T er komplementære baser, og det samme for G og C. Strukturen til DNA-kjedene som er koblet sammen gjør at dobbeltrådet DNA ofte vil kveile seg opp i en slags spiral, kalt en *dobbel heliks*. Se figur 2.1 for en skjematisk illustrasjon av en bit av et dobbeltrådet DNA-molekyl. De to trådene som er bundet sammen i heliksen kalles for *komplementære tråder*, ettersom reglene for sammenbinding av baser sikrer at den ene tråden vil være entydig gitt ut fra den andre tråden. For eksempel vil tråden A-A-G-T-G ha den komplementære tråden T-T-C-A-C. Disse to trådene i DNA-molekylet vil også ha motsatt retning, det vil si at de to trådene ligger slik:

$$\begin{array}{lll} \rightarrow & 5'\text{-A-A-G-T-G-}3' & \rightarrow \\ \leftarrow & 3'\text{-T-T-C-A-C-}5' & \leftarrow \end{array}$$

To slike tråder kalles *antiparallelle*. Når to tråder som er bundet sammen skiller lag kalles det *denaturering*. Dette kan blant annet skje ved oppvarming av DNA. *Renaturering* vil si at to enkelttråder danner en dobbeltrådet DNA igjen. Dette kan skje dersom denaturert DNA avkjøles. Sammensmelting av to tråder kalles generelt *hybridisering*. De to trådene trenger ikke ha lik størrelse eller være 100% komplementære for å hybridisere, og man kan derfor under gitte betingelser få lokale områder med dobbeltrådet DNA atskilt av områder med enkelttrådet DNA. Muligheten for at to komplementære DNA-tråder kan hybridisere er grunnlaget for mange viktige analysemetoder, blant annet mikroarrayer.

2.1.2 Gener og genuttrykk

Et *gen* er en avgrenset del av et kromosom, som kreves for produksjon av et funksjonelt produkt, for eksempel et protein. Denne biten inneholder koden for hvordan et slikt funksjonelt produkt skal lages. De fleste gener i vårt genom koder for proteiner. Proteiner er bygd opp av 20 ulike byggestener, som kalles aminosyrer. Gener som koder for proteiner inneholder oppskriften på rekkefølgen av aminosyrene som skal brukes til å bygge proteinet, og hvilken struktur proteinet skal ha. Proteiner er de utøvende enhetene i kroppen vår, i den forstand at de opptrer som strukturmolekyler, signalmolekyler og enzymer, i tillegg til at de kan ha en rekke andre funksjoner. Som signalmolekyler er proteiner blant annet med på å regulere



Figur 2.1: Skjematisk figur over oppbygningen av et dobbeltrådet DNA molekyl. Heliksen er bundet sammen mellom de komplementære basene, A mot T og G mot C.

aktiviteten til de fleste genene. Videre i denne oppgaven vil det kun bli fokusert på gener som koder for proteiner.

Alle cellene i en organisme inneholder det samme genmaterialet, men ikke alle gener er uttrykt i alle typer celler eller til enhver tid. Enkelte gener er uttrykt i alle typer celler hele tiden, mens andre gener kun er aktive i spesielle typer celler eller i spesielle stadier av organismens utvikling. At et gen *uttrykkes (is expressed)* innebærer at genkoden oversettes fra DNA gjennom et beskjedmolekyl, mRNA, til et protein. (Merk at denne betegnelsen gjelder kun for gener som koder for proteiner.) Francis Crick argumenterte i 1957 for at geners primære rolle er nettopp å produsere proteiner, og at det er rekkefølgen av baser i DNA som angir rekkefølgen av aminosyrer i et protein. Han framsatte også det som siden har blitt kjent som det sentrale dogmet i molekylærbiologi, nemlig at informasjon overføres fra DNA via RNA til proteiner, men ikke andre veien (vi vet i dag at det siste er en sannhet med modifikasjoner, men for våre formål kan vi se bort fra dette):

$$\text{DNA} \rightarrow \text{RNA} \rightarrow \text{Protein}$$

DNA er oppskriften, i form av gener, RNA er budbringeren i form av mRNA og proteinene er sluttproduktet. RNA spiller altså en viktig rolle i uttrykket av gener. Vi skal kort se på kontroll av genuttrykk før vi ser nærmere på RNA og på hvilken måte det inngår i proteinsyntesen.

Kontroll av genuttrykk

Et *uttrykt* gen i en celle kopieres som sagt fra DNA til mRNA, som igjen oversettes til proteiner som utfører mange viktige oppgaver i cellen. Hver celle i organismen benytter seg bare av deler av genmaterialet. Hvilke gener som er uttrykt i en celle bestemmer hva slags celletype den er, og gir hver celletype sine unike egenskaper. Uttrykk av gener styres av komplekse og nøye regulerte mekanismer som tillater cellen å dynamisk tilpasse seg stimuli fra miljøet rundt eller tilpasse seg endringer i egne behov. Hvilke gener som uttrykkes i en celle på et tidspunkt er med på å bestemme hvilke funksjoner cellen kan utføre, det vil si om cellen skal dele seg,

om den skal ta opp næring, om den skal endre egenskaper og liknende. Hvis det oppstår feil i reguleringen av genuttrykk kan det få store konsekvenser for cellen. Dersom for eksempel cellen mister mekanismer for kontroll av cellevekst vil cellen kunne starte å dele seg ukontrollert. Kreft er eksempel på en sykdom som skyldes feil i regulering av celledelingen og som resulterer i at noen celler får vekstfordeler i forhold til andre celler.

Studier av endringer i genuttrykk kan gi verdifull innsikt i hvordan celler fungerer på et molekylært nivå og hva som skjer når en celle endrer oppførsel fra det normale. Ved å studere syke celler vil man kunne se hva som skiller en syk celle fra en frisk celle, og muligens si noe om årsakene til ulike sykdommer. Ved å studere endringer i genuttrykk under sykdom vil man også kunne finne hvilke ekspresjonsmønstre som karakteriserer en bestemt sykdom, og dermed forbedre mulighetene til spesifikke diagnoser.

Det er mulig å gjøre slike målinger av uttrykket for tusenvis av gener i en celle ved hjelp av mikroarrayer. Før vi går inn på detaljene rundt hvordan dette kan gjøres skal vi se nærmere på egenskaper til RNA og på hvilken måte mRNA er viktig for å kunne måle genaktivitet ved hjelp av mikroarrayer.

2.1.3 RNA

RNA eller *ribonukleinsyre*, er i likhet med DNA et kjedemolekyl, men med en litt annen kjemisk sammensetning enn DNA. Sukkerenheten i RNA er litt anderledes enn i DNA, da deoksyribose i DNA er byttet ut med ribose i RNA. Basene er de samme, bortsett fra at basen T i DNA byttet ut med basen uracil (U) i RNA. Dette gjør at RNA har litt andre kjemiske egenskaper enn DNA, blant annet blir RNA som regel enkelttrådet. Den har likevel muligheter for komplementær baseparing på samme måte som DNA. Den enkelttrådede strukturen gjør at RNA er vesentlig mer ustabilt enn DNA og det har derfor en mer begrenset levetid. Det finnes tre hovedklasser RNA som er med i proteinsyntesen. Avskriften fra DNA kalles mRNA (messenger RNA). Proteinproduksjonen foregår ved at kodende DNA brukes som et *templat*, det vil si en slags oppskrift, for å danne en mRNA-tråd. Denne inneholder samme sekvensen som DNA'et som ble lest av, bortsett fra at T er erstattet med U. RNA brytes ned relativt raskt under denne prosessen, så det må produseres fortløpende. Denne mRNA-tråden brukes videre som en guide i sammenhektingen av aminosyrer til et protein. I tillegg finnes det tRNA (transfer RNA) og rRNA (ribosomal RNA) som også deltar i proteinproduksjonen på ulike stadier, men de inneholder ikke informasjon om genet på samme måte som mRNA'et.

Et mRNA-molekyl er altså et bindeledd mellom DNA og det produserte proteinet. Det er likevel ikke alltid en direkte sammenheng mellom mengden mRNA og mengden av det tilsvarende proteinet. Dette skyldes ulik hastighet på avlesning fra DNA (transkripsjon) og oversetting til protein (translasjon) og i tillegg stabiliteten til mRNA-molekylet og proteinet. Likevel brukes vanligvis mengden av mRNA til å kvantifisere genuttrykk. En viktig grunn til dette er at direkte målinger av proteinnivå krever ofte ulike metoder for ulike proteiner. Disse kan være både kompliserte og tidkrevende, og for mange proteiner eksisterer det ikke engang metoder for å kunne detektere dem. Det er derimot relativt enkelt å måle mengden av alle mulige mRNA med en og samme teknikk, nemlig DNA-mikroarrayer.

2.2 DNA-mikroarrayteknologien

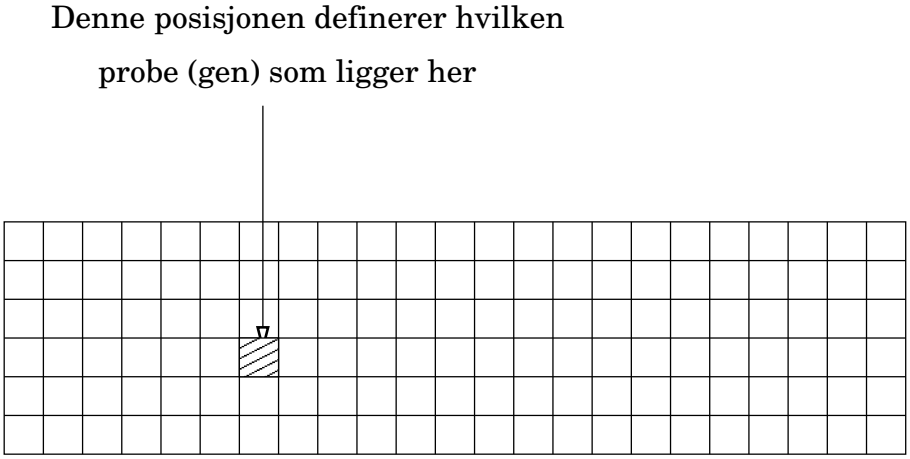
DNA-mikroarrayteknologien gjør det mulig å måle genuttrykk for et stort antall gener samtidig ved å utnytte de spesielle hybridiseringsegenskapene til komplementære DNA- og mRNA-sekvenser. Ettersom bare komplementære sekvenser vil binde seg til hverandre er det mulig å hybridisere et stort antall forskjellige komplementære nukleotidsekvenser samtidig. Mikroarrayer brukes i hovedsak til analyse av genuttrykk, og det er denne bruken vi skal se nærmere på i denne oppgaven. Man finner derimot stadig nye bruksområder for slike arrayer, de kan for eksempel brukes til både mutasjonsdeteksjon og sekvensering av DNA[46, 49].

2.2.1 Mikroarray ekspresjonsanalyse

I mikroarray ekspresjonsanalyse måler man det globale genuttrykket i celler fra en vevsprøve ved å måle mengden mRNA i vevsprøven for et stort antall gener. Resultatet er et sett av genekspresjonsmålinger $(x_1, \ldots, x_p)$, hvor p er antall gener man måler for og x_i er den målte mengden mRNA for det i'te genet. En høy verdi for x_i betyr at prøven inneholdt mye av transkriptet (mRNA'et) for det i'te genet, og følgelig at det i'te genet er høyt uttrykt i vevsprøven. For å utføre en mikroarray ekspresjonsanalyse må man først isolere alt RNA (*total RNA*) fra vevsprøven. Det isolerte mRNA'et er ustabilt og blir derfor lett degradert. For å kunne bruke mRNA i ekspresjonsstudier vil man gjerne overføre det til en mer stabil form. Dette kan gjøres ved en metode som kalles *revers transkripsjon*. Denne metoden går kort sagt ut på at mRNA oversettes tilbake til DNA som er mer stabilt enn mRNA, ved hjelp av et enzym som kalles revers transkriptase. Produktet av denne metoden er en sekvens med DNA som er komplementær til mRNA-sekvensen, og kalles derfor *komplementær DNA (cDNA)*.

Hvert gen har mRNA som er spesifikt for dette bestemte genet, og slikt mRNA kan detekteres ved hjelp av å hybridisere dem til nukleotidsekvenser som er festet til en *mikroarray*. En mikroarray er en liten glassplate eller membran på størrelse med en mikroskopslide. Disse nukleotidsekvensene er arrangert i et regulært matrisemønster på denne platen. Mønsteret består av et visst antall punkter (spots), der hvert punkt inneholder nukleotidsekvenser som tilsvarer et spesifikt gen. Nukleotidsekvensene som festes på arrayen kalles for *prober*. Probene brukes til å detektere mRNA fra et gen og velges slik at det inneholder en unik sekvens som er komplementær til det genet man ønsker å detektere. mRNA reverstranskriberes til cDNA, slik at probene må være komplementære til genet, det vil si like som mRNA-sekvensen. I praksis lages dobbeltrådet cDNA fra mRNA, som igjen hybridiseres mot dobbeltrådet cDNA i probene etter denaturering. Hver probe inneholder et stort antall identiske nukleotidsekvenser, slik at så stor mengde cDNA man ønsker kan hybridisere til probene. Det er viktig at probene sitter fast til underlaget i et bestemt mønster, ettersom hvert enkelt punkt brukes til å identifisere et spesielt gen, se figur 2.2. Det kan være tusenvis av slike prober på en plate, tilsvarende antall gener man ønsker å studere.

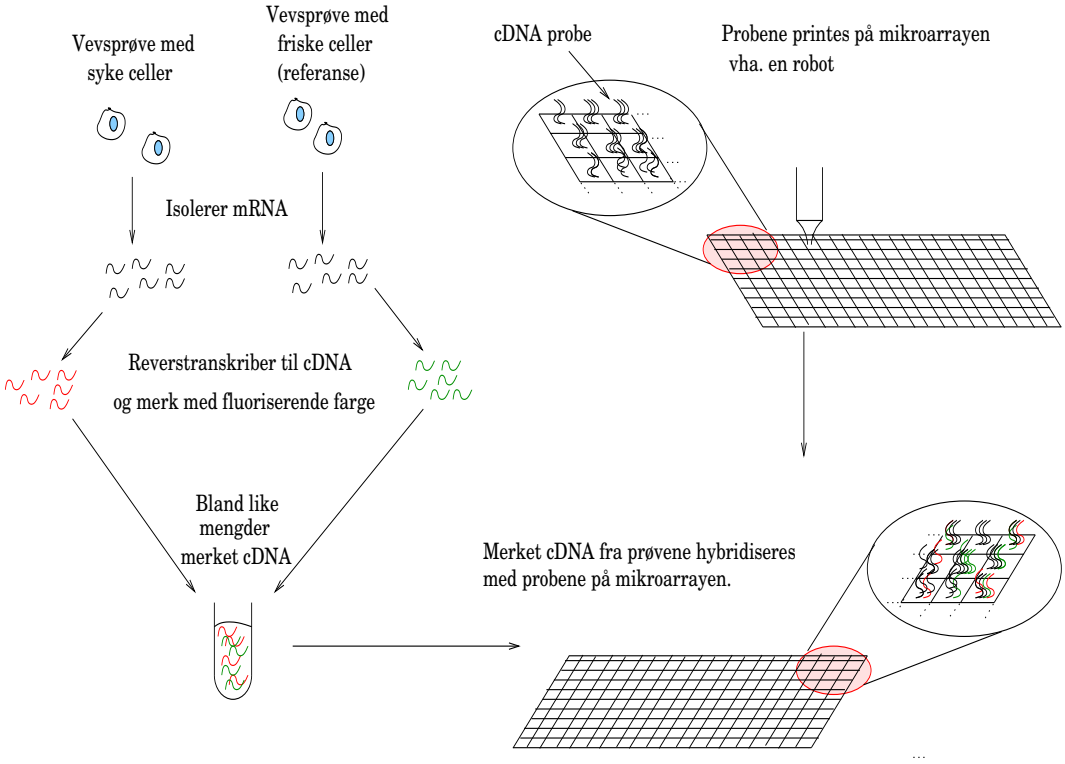
Det er flere typer mikroarrayer som er vanlige å bruke i ekspresjonsanalyse. Forskjellen mellom dem er blant annet hvordan man lager og fester probene til arrayen som skal brukes i eksperimentet. En type arrayer kalles *oligonukleotidarrayer* [35] og ble først fremstilt av Affymetrix. Slike arrayer inneholder par av prober for hvert mRNA som overvåkes. Hvert probepar består av to korte nukleotidsekvenser, oligonukleotider, designet slik at de hybri-



Figur 2.2: En illustrasjon av en mikroarray. Hver probe festes på arrayen i en definert posisjon som senere kan brukes til å identifisere dette genet. Antall posisjoner i figuren er $6 \times 23 = 138$. I praksis er tallet mye høyere, for eksempel 48.000.

diserer til ulike deler av mRNA- (eller mer presist cDNA-) molekyler. Den ene sekvensen i paret er perfekt komplementær til en unik subsekvens av genet vi ønsker å studere (perfect match probe). Den andre sekvensen i paret er identisk, bortsett fra at en base er byttet ut i en sentral posisjon i proben (mismatch probe). Denne mismatch proben i hvert par tjener som en intern sikkerhetskontroll mot feilhybridisering. På disse arrayene blir sekvensene som skal studeres syntetisert rett på arrayen, ved hjelp av en teknikk som kalles *foto-litografi*. En annen type arrayer kalles *cDNA-mikroarrayer*. Probene på arrayen består i dette tilfellet av cDNA som er utledet fra mRNA-sekvensen til kjente gener i den vevstypen som studeres. Disse kan finnes i et såkalt *cDNA-bibliotek* som er en samling av cDNA-sekvenser til kjente gener. Disse sekvensene kan kopieres opp i stort antall og festes på arrayen ved hjelp av en robot. En tredje type mikroarrayer som det knyttes store forventninger til er fiberoptiske arrayer, hvor et stort antall (opptil millioner) gener kan analyseres på en gang, og disse er svært sensitive. Dette gjør det mulig å studere ulike varianter av gener som kan oppstå på grunn av ulik mRNA-spleising (det kan oppstå mange forskjellge proteiner fra ett enkelt gen på grunn av ulik spleising av mRNA produktet) og det er mulig å studere biter av gener, det vil si at man kan studere det samme genet ved å bruke ulike prober som er karakteristiske for akkurat dette genet. [14, 53, 55]

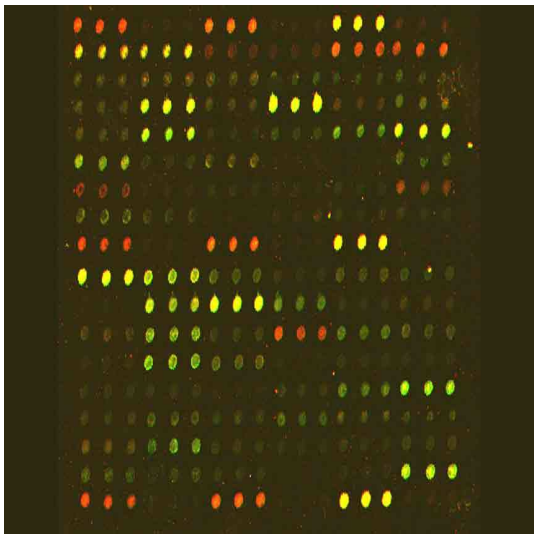
Mikroarrayer brukes i hovedsak til å detektere endringer av genuttrykk ved å sammenlikne to kilder. Dette gjøres for eksempel ved å sammenlikne prøver fra to syke pasienter som har mottatt ulik behandling, eller ved å sammenlikne prøver fra syke pasienter mot prøver fra en frisk person. En slik sammenlikningsstudie gjøres ved å først hente ut mRNA fra to celleprøver, en fra den syke celletypen og en fra den friske celletypen. Prøven fra den friske celletypen kalles gjerne for en *referanse*. Videre konverterer man mRNA fra begge kildene til dobbeltrådet cDNA ved hjelp av revers transkriptase. Hver prøve er da en blanding av mange ulike typer cDNA som tilsvarer sammensetningen av mRNA-blandingen i det vevet vi isolerte mRNA'et fra. Deretter merkes cDNA fra de to kildene med hver sin fluorescerende farge. Referansen merkes vanligvis med grønn farge, mens prøven fra de syke cellene merkes med rød farge. Etter merkingen blandes like mengder av begge prøvene til en løsning, som så



Figur 2.3: Skisse av et mikroarrayforsøk.

varmes opp for å denaturere dobbeltrådet cDNA og helles over en mikroarray for å hybridisere. På mikroarrayen har man allerede festet prober for de genene som man ønsker å studere. Dersom det finnes en nukleotidsekvens i en av probene på arrayen som er komplementær til en cDNA sekvens i en av prøvene, vil disse binde seg til (hybridisere med) hverandre. cDNA fra prøvene blir altså sittende fast på platen på det punktet der det finnes en probe med en nukleotidsekvens som er komplementær med denne cDNA-sekvensen. Ettersom hver probe inneholder svært mange nukleotidsekvenser vil det være mulig for cDNA fra begge prøvene å feste seg uten å måtte konkurrere om plassen. Etter hybridiseringen vaskes materiale som ikke har festet seg til platen vekk. En probe vil da inneholde rødt/grønt fargestoff dersom prøven merket med rødt/grønt inneholdt mRNA fra genet som tilsvarer proben og dermed har festet seg på dette punktet. Figur 2.3 viser en skisse over denne prosessen.

Den fluorescerende fargen gjør at det er mulig å måle hvor mye cDNA fra syke prøven og fra referansen som har festet seg til hvert punkt på mikroarrayen. Dette gjøres ved å scanne arrayen, det vil si at den utsettes for stråling fra en laser som gjør at det fargemerkede cDNA som har festet seg vil fluorescere. Man måler da intensiteten til begge fargene i hvert punkt. For å oppnå informasjon om genekspresjonsnivå må slike scanninger analyseres. Intensiteten til de ulike punktene må måles, avgrenses, justeres med hensyn på med bakgrunnen og liknende. Dette kalles bildekvantifisering og gjøres ved bruk av software for bildeanalyse[2]. Detaljene rundt dette vil ikke blir gjennomgått her. Resultatet av en bildeanalyse er at intensiteten til hver av de to fluorescerende fargene i hvert punkt blir målt og justert slik at målingene vil korrespondere med total mengde hybridisert merket cDNA i dette punktet, som igjen korresponderer til total mengde av den tilsvarende typen mRNA i prøvene.

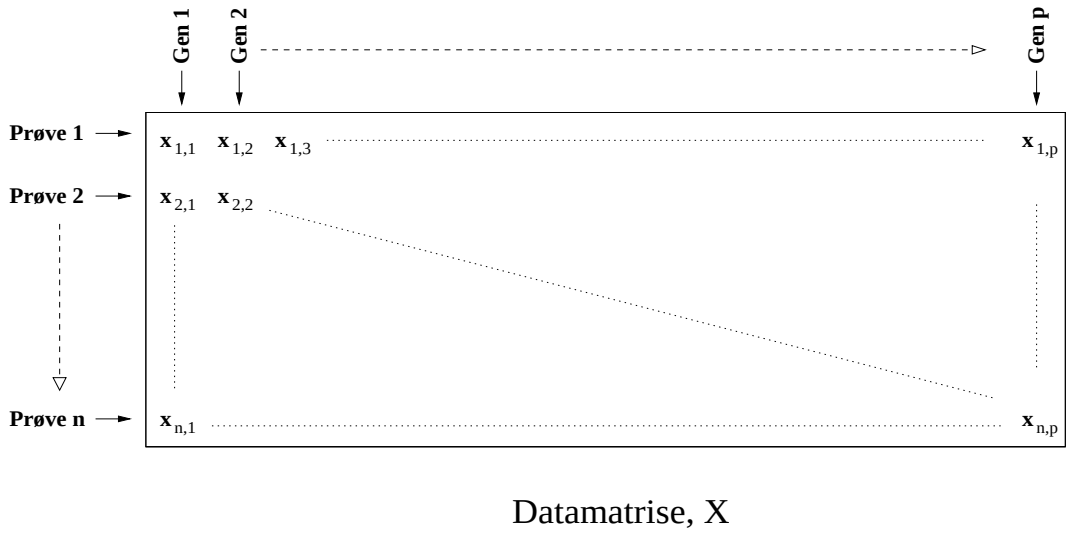


Figur 2.4: Bilde av en scannet mikroarray. Rød farge tilsvarer oppregulert gen i syk prøve, mens grønn farge tilsvarer nedregulert gen i syk prøve. Punkter med gul farge tilsvarer lik regulering i begge prøver, mens svarte punkter angir ingen ekspresjon av dette genet i noen av prøvene.

Scanninger av mikroarrayer vises ofte som et bilde, der hvert punkt på arrayen representeres med en farge. Denne fargen tilsvarer hvor mye mRNA det var i de ulike prøvene. Punktet vil bli grønt dersom prøven merket med grønt hadde høyere konsentrasjon av mRNA som tilsvarer genet i dette punktet, enn prøven som ble merket med rødt. Punktet blir rødt i det motsatte tilfellet. Like mengder mRNA for et gen i begge prøvene vil føre til et gult punkt, mens hvis ingen av prøvene inneholdt mRNA for et bestemt gen vil punktet bli sort, se figur 2.4. Intensiteten for hvert punkt oppgis ofte som ratioen mellom intensiteten av de to fargene. Dette kommer av at man kan få mer presise målinger av forholdet mellom uttrykket i de to prøvene, enn av et absolutt signal alene. Ratioen mellom den grønne og den røde fargen bestemmes og vil gi en indikasjon på om nivået på genuttrykket i den syke celleprøven er opp- eller nedregulert i forhold til referansen. Grunnen til at man ofte velger å se på ratioen mellom de to prøvene er at man på denne måten kan fjerne en del systematiske feilkilder, for eksempel multiplikative feilkilder som er felles for både rød og grønn kanal. For eksempel kan det på grunn av ulike mengder materiale i probene feste seg ulik mengde cDNA fra prøvene. Mengden rødt og grønt materiale som fester seg i en probe antas å være proporsjonal med mengde mRNA i henholdsvis den røde og grønne prøven. Disse proporsjonalitetsfaktorene antas å være identiske for rød og grønn, men også genspesifikke. Dette vil si at det i en probe kan feste seg tre ganger så mye materiale som i en annen probe. Ved å se på ratioen i mellom signalene i hvert punkt fjerner man altså denne typen ulikheter mellom punkter.

2.3 Data fra mikroarrayer

Mikroarrayforsøk produserer store mengder data. Hver celleprøve kjøres på hver sin array, gjerne mot den samme referansen. For hver slik array lages en så genekspresjonsvektor, det



Figur 2.5: Illustrasjon av et datasett. Hver rad representerer ekspresjonen av genene i ett forsøk, mens hver kolonne representerer ekspresjonen til ett bestemt gen over alle forsøkene. Posisjon $x_{i,j}$ representerer ratioen av den målte intensiteten for gen j i prøve i.

vil si en vektor med de beregnede ratioene mellom rødt og grønt signal for hvert av de studerte genene. Slike vektorer settes sammen til en matrise slik at hver rad i matrisen er genekspresjonsvektoren for en celleprøve og hver kolonne er ekspresjonsvektoren for et gen. Se figur 2.5 for en illustrasjon av et datasett fra mikroarrayer. Dersom $x_{i,j} > 1$ sier man at gen j i prøve i er *oppregulert* i forhold til referansen. I motsatt tilfelle sier man at det er *nedregulert*.

Et datasett fra mikroarrayforsøk vil vanligvis bestå av få observasjoner eller prøver, n, og målinger for et stort antall gener p. En slik tabell med genekspresjonsdata er utgangspunktet for statistisk analyse av mikroarraydata. Før en slik analyse er det vanlig å preprosessere rådataene på ulike måter.

2.3.1 Preprosessering av data

Det er vanlig å preprosessere data på en eller flere måter. Hva som gjøres varierer i ulike situasjoner og hva slags type analyser man har tenkt å utføre. Vi skal se kort på et par av de vanligste metodene som finnes.

En vanlig ting å gjøre er å fjerne gener som har tilnærmet helt lik ekspresjon over alle prøvene. Ettersom ekspresjonen av disse genene er nesten helt lik for alle prøvene vil de ikke kunne bidra til å kunne skille de ulike prøvene fra hverandre, og denne informasjonen vil derfor bli overfladisk. Ved å fjerne slike gener vil man kunne redusere støymendgen i dataene.

Videre vil man kunne oppleve at det er en del verdier som mangler i datasettet. Dette kommer av at det er en del usikre målinger fra arrayen som man har måttet utelukke. Dette kan skyldes utflytende punkter, feil ved arrayen og liknende. For å håndtere problemet med manglende verdier kan man gjøre en av følgende:

- bare inkludere gener som har verdier i videre analyse,
- imputere manglende verdier, eller
- bruke analysemetoder som tillater manglende verdier.

Det å fjerne alle gener med manglende verdier er som regel en dårlig strategi, ettersom man da må kaste mye av datamaterialet. Imputasjon er en bedre strategi som går ut på at man beregner de manglende verdiene ut fra verdier i de andre prøvene. Dette kan for eksempel gjøres ved å ta et gjennomsnitt av alle, ved å ta gjennomsnitt av de K prøvene som har mest liknende genuttrykk (K-nærmeste nabo) eller ved andre mer sofistikerte metoder. Stort sett er det ulike imputasjonsmetoder som benyttes for å håndtere dette problemet.

Neste skritt er vanligvis å log_2-transformere de målte intensitetsratioene i datamatrisen. Dette gjør at man får en lik skala for både opp- og nedregulering av uttrykket. Vi ser dette ved at dersom uttrykket er likt for både prøven og referansen blir ratioen 1, mens log_2-ratioen blir 0. Dersom genuttrykket er dobbelt så høyt i prøven som i referansen vil log_2-ratioen bli 1, mens halvparten så høyt vil resultere i en log_2-ratio på -1. Et gen er overuttrykt i forhold til referansen dersom log_2-ratioen er over 0 og underuttrykt dersom den er mindre enn 0. En slik transformasjon kan rettferdiggjøre en implisitt antagelse om normalfordelte data og kan med det lette modelleringen av dataene[24].

Etter en slik transformasjon er det vanlig å normalisere dataene. Hensikten med dette er å fjerne systematiske feil som kan komme av at de ulike fluorescerende fargene virker litt forskjellig, for eksempel at forsøket ble utført med ulike mengder rødt og grønt merket cDNA, uregelmessigheter i selve arrayen eller liknende. Ved å normalisere dataene ønsker man at enhver forskjell i målt intensitet skal skyldes ulikt uttrykk av genene i prøvene. En normalisering gjøres normalt før en analyse, men det kan også inngå som en del av analysen. Det finnes to hovedtyper normalisering. Den første kalles skalering og sikrer at variansen til hvert gen er lik, vanligvis settes den til 1. Den andre typen kalles sentrering og sikrer at alle genene har likt gjennomsnitt. Vanlig å bruke er et gjennomsnittlig målt intensitetsratio på 1. Disse to måtene å normalisere på endrer ikke på selve datamaterialet i forhold til analyse, men det gjør det blant annet lettere å sammenlikne eksperimenter over flere arrayer. Det finnes en rekke andre metoder for å korrigere for andre typer systematiske feil, men disse vil ikke bli gjennomgått her. For nærmere detaljer se [43].

2.3.2 Analysemetoder

Data fra mikroarrayforsøk inneholder mye støy som oppstår under de ulike stegene i prosessen ved produksjon av dataene, for eksempel ved produksjon av matrisene, fra hybridiseringen og fra scanningen. Statistiske analysemetoder bør ta hensyn til denne støyen på best mulig måte. Målet med å analysere dataene er å klare å hente ut relevant biologisk informasjon fra dataene, til tross for støyen. Det eksisterer en rekke ulike analysemetoder for mikroarraydata. De deles ofte inn i to hovedgrupper, kalt *ikke-overvåkede* og *overvåkede* analysemetoder.

Med ikke-overvåkede metoder menes at kun dataene fra mikromatrisen brukes i analysen, ingen annen utenforliggende informasjon. Eksempler på dette er ulike klustrings- og klassifiseringsmetoder og multippel hypotesetesting for å teste hvilke gener som har signifikant ulikt

uttrykk. Slike metoder er svært populære til analyser av mikroarraydata. Se [12] for nærmere informasjon om slike metoder. Overvåkede metoder bruker derimot annen tilgjengelig informasjon om prøvene og/eller genene som studeres i tillegg til ekspresjonsdataene. Man kan for eksempel bruke målinger for ulike kliniske variable for individene i studien til å gjøre en overvåket diskriminantanalyse, eller man kan utføre regresjonsanalyser der man har en respons som man ønsker å relatere til genekspresjonsdataene.

Ulike analyseteknikker fokuserer på ulike aspekter ved dataene, og vil derfor kunne hente ut ulik informasjon fra det samme datasettet. Det er derfor nødvendig å analysere de samme datasettene med ulike metoder. For å få et inntrykk av hvor god en metode er, vil det også være avgjørende å teste den ut på mange ekspresjonsdatasett.

2.4 Bruk av mikroarrayer

Helt til slutt i dette kapittelet skal vi se på noen av bruksområdene til mikroarrayer og litt på hva ekspresjonsanalyse kan bidra med.

Ekspresjonsanalyse brukes, som allerede nevnt, i hovedsak til å studere endringer i genuttrykk ved sykdom og for å kunne studere vekst og utvikling av en organisme. En av de viktigste bruksområdene er å kunne forstå de underliggende genetiske årsakene til mange typer sykdommer. Kreft er en av de vanligste sykdommene som er relatert til endringer i genmaterialet. Kreft kan skyldes mutasjoner i DNA-sekvensen som oppstår for eksempel på grunn av feilkopiering av DNA under celledeling, eller når cellen utsettes for ytre påvirkninger som kan skade DNA'et. Mikroarraystudier av ulike krefttyper har blant annet gjort at man har oppdaget ulike undertyper av den samme krefttypen[50]. Slike undertyper av kreft har ikke vært mulig å oppdage med klassiske diagnostiseringsmetoder, ettersom forskjellene mellom dem bare kan identifiseres på et molekylært nivå. Ved å studere ekspresjonen av gener har det derimot vært mulig å identifisere flere ulike undergrupper. En slik oppdagelse er viktig, ettersom det viser seg at pasienter som lider av ulike undertyper av den samme typen kreft kan ha ulike kliniske forløp. Prognoser og behandlingsrespons vil kunne variere mellom de ulike undertypene, så pasienten vil kunne bli gitt en mest mulig optimal behandling ved å bli diagnostisert med riktig undertype av kreften.

Ved å detektere genetiske endringer vil det i mange sykdomstilfeller også kunne være mulig å oppdage sykdommen og gi en sikrere diagnose på et tidligere stadium, enn ved å benytte klassiske diagnosemetoder. Dette vil gi muligheten til å starte en optimal behandling så tidlig som mulig i sykdomsforløpet, og med det muligens bedre pasientens sjanse til å bli frisk eller til et bedre og lengre liv. Ved å kjenne en pasients totale ekspresjonsprofil vil det også være mulig å gi behandlinger som er optimal for hver enkelt person. Dette er blant annet et viktig tema innen legemiddelindustrien[49].

Mikroarrayteknologien er stadig under utvikling og den vil uten tvil bli svært viktig også i årene som kommer. Selv om teknologien har kommet langt er det enda en stor utfordring i å finne gode metoder for å analysere dataene. Bioinformatikk er en viktig bidragsyter til utviklingen av gode og effektive metoder for analyse av slike data.

Kapittel 3

Overlevelsesanalyse

3.1 Innledning

Som nevnt i innledningen er overlevelsesanalyse i ferd med å bli et viktig felt innenfor bioinformatikk. Overlevelsesanalyse er en retning innen statistikk som studerer det som kalles *overlevelsestider* eller *ventetider*. En slik overlevelses- eller ventetid defineres som tiden det tar før en definert hendelse inntreffer. Eksempler på dette kan være tiden det tar fra en person blir født til vedkommende dør, tiden det tar til en komponent i en elektrisk krets feiler eller tiden det tar fra en pasient mottar en behandling til vedkommende får et tilbakefall i sykdommen. Dersom man studerer hendelser som kan inntreffe mer enn en gang kalles det *forløpsanalyse*. Er det derimot tiden fram til en enkelt hendelse man fokuserer på, kalles det overlevelsesanalyse. Overlevelsesanalyse er altså et delområde av forløpsanalysen. Betegnelsen kommer fra at de første statistiske metodene innenfor dette feltet ble utviklet for å studere dødelighet[1]. De samme metodene kan brukes til å studere en hvilken som helst type hendelse, men ofte er det dødsfall eller svikt man er interessert i. Hensikten med å gjøre slike statistiske analyser er blant annet å finne ut hvilke faktorer som er prediktive for levetiden, for eksempel om en behandling kan forlenge levetiden til en person eller for eksempel om det at man røyker eller ikke spiller inn på levetiden.

Felles for alle typer overlevelsesdata er et fenomen som kalles *sensurering*, et viktig begrep innen overlevelsesanalyse. Det er vanlig å skille mellom to typer sensurering. (Det eksisterer også flere typer sensurering [28] med disse vil ikke bli diskutert her.) Den første typen oppstår når pasienter taes opp i studien, etter at hendelsen har inntruffet. Da vet man at personen har opplevd hendelsen, men det eksakte tidspunktet for hendelsen er ukjent. Som eksempel kan tas en pasient som har blitt operert for kreft. Tre måneder etter operasjonen taes vedkommende opp i en studie som undersøker tilbakefall, og man oppdager da at pasienten allerede har opplevd et tilbakefall. Da er tidspunktet for tilbakefallet ukjent, det eneste man vet at det inntraff mindre enn 3 måneder før opptak i studien. Denne typen sensurering kalles for *venstresensurering*.

Den andre typen sensurering kalles *høyresensurering*. Denne typen sensurering innebærer at man vet at hendelsen ikke har inntruffet innen en viss tid, typisk innen studien avsluttes.

Pasient nr.	Tid (måneder)	Sensurstatus	Behandling	Kjønn	Alder (år)
1	0.5	1	1	1	54
2	3.1	0	1	1	35
3	24.0	0	0	0	67
4	7.2	1	1	0	38
5	23.8	1	1	0	79
6	3.4	0	0	1	62
7	8.4	1	0	1	59
8	15.9	1	0	1	45
9	24	0	0	1	24
10	9.9	1	1	0	81

Tabell 3.1: Et eksempel på et datasett med overlevelsesdata. For hver pasient er levetid, sensurstatus og de tre kovariatene behandling, kjønn og alder registrert.

Man vet derfor ikke eksakt tid for hendelsen, bare at den vil inntreffe en gang etter man har sluttet å observere. Det kan være flere årsaker til høyresensurering. Det er for eksempel vanlig at pasienter rekrutteres til en studie over tid, og det er derfor ikke sikkert at alle pasienter vil oppleve hendelsen innen studien avsluttes. Andre årsaker kan være at noen av ulike grunner velger å forlate studien underveis, eller at personer dør av andre årsaker, såkalte *konkurrerende hendelser*. Ved statistiske analyser av høyresensurerte data er det viktig å ta hensyn til om det har skjedd *selektiv høyresensurering*. Det skjer dersom de som faller ut av studien skiller seg ut på noen systematisk måte i forhold til de som fortsetter i studien. Dersom for eksempel mange slutter i en behandlingsstudie på grunn av bivirkninger ved behandlingen, uten at man kartlegger årsaken til at man trekker seg fra studien, vil man kunne komme til å dra uriktige konklusjoner fra analysen senere. Man må altså ta dette med i vurderingen når man skal analysere data. De dataene som vil bli brukt senere i oppgaven har ikke venstresensurering eller selektiv sensurering, og det vil bli antatt i det følgende at ingen av delene forekommer. I fortsettelsen menes derfor høyresensurering når det er snakk om sensurering.

For hvert individ i en overlevelsesstudie har man registrert en tid t, som tilsvarer tiden det tar fra man starter å observere til individet opplever enten en hendelse eller blir sensurert. For hvert individ har man også registrert en sensurindikator δ, slik at $\delta = 0$ dersom individet ble sensurert ved tid t og $\delta = 1$ dersom individet opplevde en hendelse ved tid t. For hvert individ kan man også ha observert en eller flere andre størrelser (kovariater) som registreres i en (kovariat-)vektor $\mathbf{x}$. I og med at man har med sensurerte observasjoner i data, må man benytte spesielle statistiske metoder for å analysere dem. Vi skal se på et lite eksempel på et slikt datasett.

Dersom man har data for n individer, registrerer man altså for hvert individ $i = 1, \ldots, n$ et tidspunkt t_i, en sensurstatus δ_i som indikerer om individet hadde en hendelse eller ble sensurert og eventuelt en eller flere kovariater som samles i vektoren $\mathbf{x}_i$. Datasettet består altså av tripletter på formen:

$$data_j = (t_j, \delta_j, \mathbf{x}_j) \quad \text{for} \quad j = 1, \ldots, n.$$

Som et eksempel kan vi tenke oss en undersøkelse med ti kreftpasienter. Man ønsker i undersøkelsen å finne ut om en ny type behandling kan forlenge levetiden i forhold til en eksisterende behandling. Pasientene randomiseres inn i to like store grupper, en gruppe som får den nye behandlingen og en gruppe som blir gitt den gamle behandlingen. I tillegg til informasjonen om behandling registreres også alder og kjønn for hvert individ. Disse tre verdiene, behandling, kjønn og alder, er da kovariatene og disse lagres i kovariatvektorer på formen $\mathbf{x}_j = (\text{behandling}_j, \text{kjønn}_j, \text{alder}_j)$ for alle pasienter j. Deretter starter pasientene behandlingen og de blir fulgt opp i en viss periode. Underveis i studien registrerer man hendelser, det vil si sensureringer og dødsfall, og tidspunktene for dette. Et eksempel på et slikt datasettet sees i tabell 3.1. Kovariaten for kjønn kodes som 0 for mann og 1 for kvinne, mens kovariaten for behandling kodes som 0 for den gamle behandlingen og 1 for den nye. For eksempel ser man av tabellen at pasient 5 dør etter 23.8 måneder etter behandlingen starter, mens pasient 2 sensureres 3.1 måneder etter at behandlingen startes.

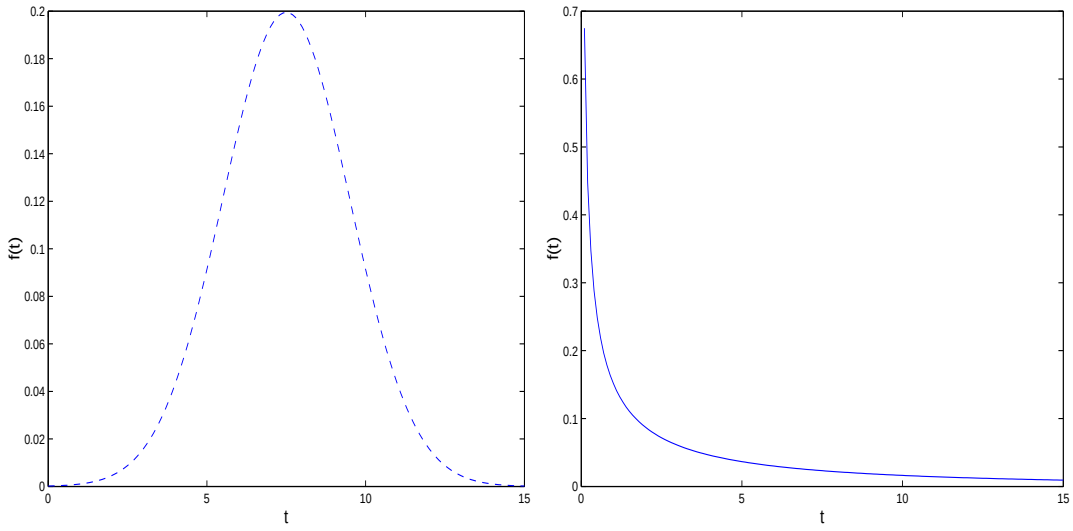
En klassisk tanke innen statistikken er at man ønsker å spesifisere en fordeling for overlevelsestiden T til et individ. Denne fordelingen kan beskrives som en hvilken som helst generell fordeling, ved for eksempel å spesifisere sannsynlighetstetthetsfunksjonen f til fordelingen. Den enkleste måten å spesifisere denne fordelingen på er å tenke på de ulike overlevelsestidene, T_i for $i = 1, \ldots, n$, som identisk fordelte stokastiske variable, det vil si realiseringer av den samme stokastiske variabelen T. Dette er det samme som å modellere overlevelse for en homogen befolkning. Vi skal starte med å se på ulike måter å spesifisere fordelingen til T på under disse forutsetningene.

Mer interessant vil være å spesifisere fordelingen til de enkelte levetidene T_i slik at de avhenger av ulike kovariater, $\mathbf{x}_i$, assosiert med individ i. Dette vil være en naturlig måte å spesifisere fordelingene til overlevelsestider på, ettersom man stort sett studerer heterogene befolkninger. Når man tar hensyn til kovariatene antar man ikke lenger at alle T_i er identisk fordelte, og metodene for å spesifisere fordelingene til T_i vil bli mer kompliserte. Vi kommer tilbake til metoder for å gjøre dette i avsnitt 3.6.

3.2 Måter å spesifisere fordelingen til levetiden T

For å måle overlevelsestider må man først definere hva som menes med tid 0. Dette kan for eksempel være tidspunktet når man blir tatt opp i studien, når man blir født, tidspunkt for diagnose av sykdom, tidspunkt for behandlingsstart eller liknende. Tid 0 trenger ikke være den samme i kalendertid for alle individer i samme undersøkelse, ettersom individene kan rekruteres over tid. La nå T være tiden det tar før en spesifisert hendelse inntreffer for ett individ. Da er T en ikke-negativ stokastisk variabel, hvis fordeling kan karakteriseres entydig ved fire begreper [28, 25]:

- Sannsynlighetstetthetsfunksjonen $f(t)$ er den ubetingede sannsynligheten for at hendelsen skal inntreffe ved en tid t
- Hasard funksjonen, eller hasardraten, $h(t)$ gir sannsynligheten for at et individ skal oppleve hendelsen i neste øyeblikk
- Den kumulative hasarden, $H(t)$



Figur 3.1: Til venstre sees tetthetsfunksjonen til en normalfordeling med parametere $\mu = 7.5$ og $\sigma = 2$ og til høyre tetthetsfunksjonen til en Weibullfordeling med parametere $\kappa = 0.5$ og $\tau = 0.5$.

- Overlevelsesfunksjonen $S(t)$ gir sannsynligheten for at en person skal overleve til etter tid t.

I og med at det finnes enkle matematiske sammenhenger mellom disse funksjonene, kan man dersom man kjenner den ene også finne de andre. Vi vil her bare se på de tilfellene der T er en kontinuerlig variabel. Det finnes tilsvarende definisjoner dersom T er diskret; de vil ikke bli omtalt her.

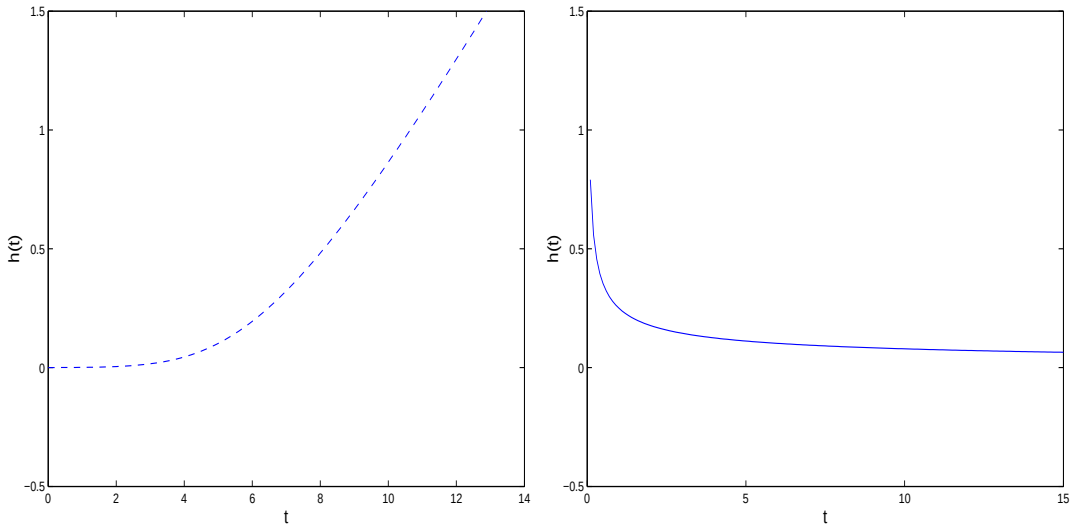
3.2.1 Tetthetsfunksjonen

Tetthetsfunksjonen $f(t)$ til en levetid T er definert som en vanlig tetthetsfunksjon, det vil si at den er ikke-negativ og arealet under summerer opp til 1. Uttrykket $f(t)\Delta t$ kan sees på som den approksimative sannsynligheten for at hendelsen til et individ skal inntreffe i et lite tidsintervall $[t, t + \Delta t)$. Den kumulative tetthetsfunksjonen $F(t)$ gir sannsynligheten for at en hendelse skjer før eller ved t, og er definert som $F(t) = P(T \leq t) = \int_0^t f(u)\mathrm{d}u$. Se figur 3.1 for eksempler på plott av to tetthetsfunksjoner.

3.2.2 Hasardraten

Hasardraten er definert som en betinget sannsynlighet. For hvert tidspunkt t finner man sannsynligheten for å oppleve hendelsen innenfor et lite intervall $[t, t + \Delta t)$, gitt at man ikke har opplevd hendelsen ved begynnelsen av intervallet. Hasardraten er da definert som denne sannsynligheten, delt på lengden av intervallet, og kan uttrykkes ved formelen

$$h(t) = \lim_{\Delta t \to 0^+} \frac{P(t \leq T < [t + \Delta t) \mid T \geq t)}{\Delta t}. \tag{3.1}$$



Figur 3.2: Til venstre sees hasarden til en normalfordeling med parametere $\mu = 7.5$ og $\sigma = 2$ og til høyre hasarden til en Weibullfordeling med parametere $\kappa = 0.5$ og $\tau = 0.5$. Disse tilsvarer tetthetsfunksjonene i figur 3.1.

Merk at $h(t)\Delta t$ er den approksimerte sannsynligheten for at et individ opplever hendelsen i neste intervall $[t + \Delta t)$, gitt at det var i live ved tid t, det vil si en øyeblikkelig risiko.

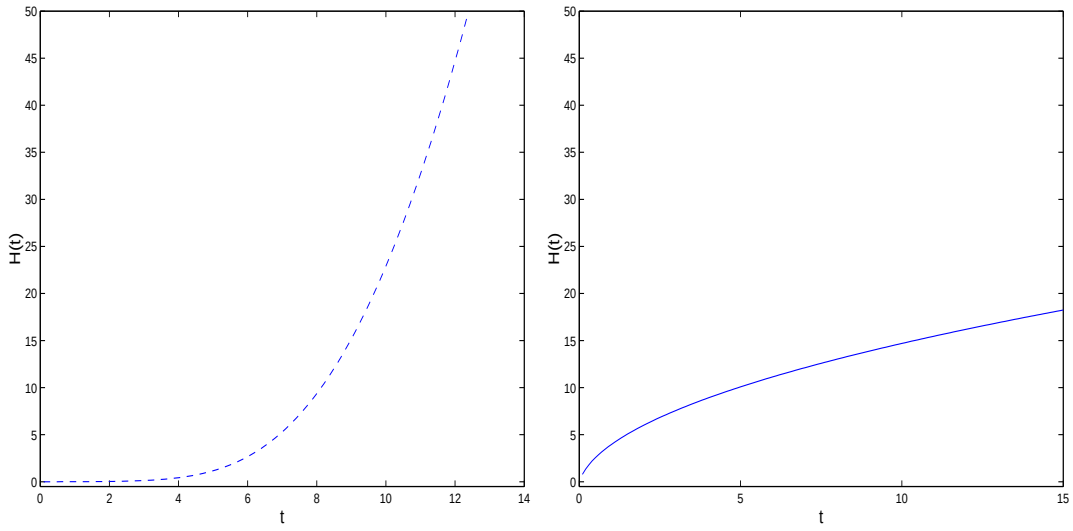
En hasardfunksjon kan ha ulike former, den eneste restriksjonen er at den må være større eller lik 0 for alle t. Det mest vanlige er å ha en hasardfunksjon som enten er konstant eller som øker med tiden, det vil si at risikoen for å oppleve en hendelse øker etterhvert som tiden går. Se figur 3.2 for to eksempler på hasardfunksjoner. Disse hasardfunksjonene tilsvarer tetthetsfunksjonene i figur 3.1.

3.2.3 Den kumulative hasarden

En størrelse som er nært relatert til hasarden er den kumulative hasarden $H(t)$, definert som

$$H(t) = \int_0^t h(u)\mathrm{d}u.$$

En måte å tolke den kumulative hasarden på er å se på den som antall hendelser som kunne vært forventet for hvert individ opp til tid t, dersom hendelsen var en repeterbar prosess. Sagt på en annen måte utgjør $H(t)$ den samlede risikoen mellom tid 0 og tid t. Den kumulative hasarden brukes ofte som et mellomsteg for å finne hasarden, men kan være nyttig for å kontrollere forutsetninger for en del analysemetoder, for eksempel proporsjonale hasarder (dette kommer vi tilbake til i avsnitt 3.9)[3]. Figur 3.3 viser plott av to kumulative hasarder som tilsvarer hasardfunksjonene i figur 3.2.



Figur 3.3: Til venstre sees den kumulative hasarden til en normalfordeling med parametere $\mu = 7.5$ og $\sigma = 2$ og til høyre den kumulative hasarden til en Weibullfordeling med parametere $\kappa = 0.5$ og $\tau = 0.5$. Disse tilsvarer hasardfunksjonene i figur 3.2.

3.2.4 Overlevelsesfunksjonen

Overlevesesfunksjonen $S(t)$ gir sannsynligheten for at et individ overlever fram til en gitt tid t, det vil si at individet opplever hendelsen etter tid t. $S(t)$ er definert som komplementet til den kumulative tetthetsfunksjonen, $S(t) = 1 - F(t) = P(T > t)$. For overlevelsesdata er $S(t)$ ofte lettere å jobbe med enn den vanlige fordelingsfunksjonen, $f(t)$. $S(t)$ er en avtagende funksjon av t som er lik 1 ved tid 0 og 0 når tiden går mot uendelig. I figur 3.4 kan man se eksempler på plott av ulike overlevelsesfunksjoner som tilsvarer hasardfunksjonene i figur 3.2 og tetthetsfunksjonene i figur 3.1. Legg merke til at selv om hasardfunksjonene er veldig ulike er det ikke alltid like lett å se dette ut fra overlevelsesfunksjonene, som alle har en synkende form. Det er ofte tilfellet at det er vanskelig å se klare mønstre ut fra kun et plott av overlevelseskurven. Den er likevel et populært mål for overlevelse i litteraturen[28], og gjør det mulig å sammenlikne to eller flere overlevelsesmønstre.

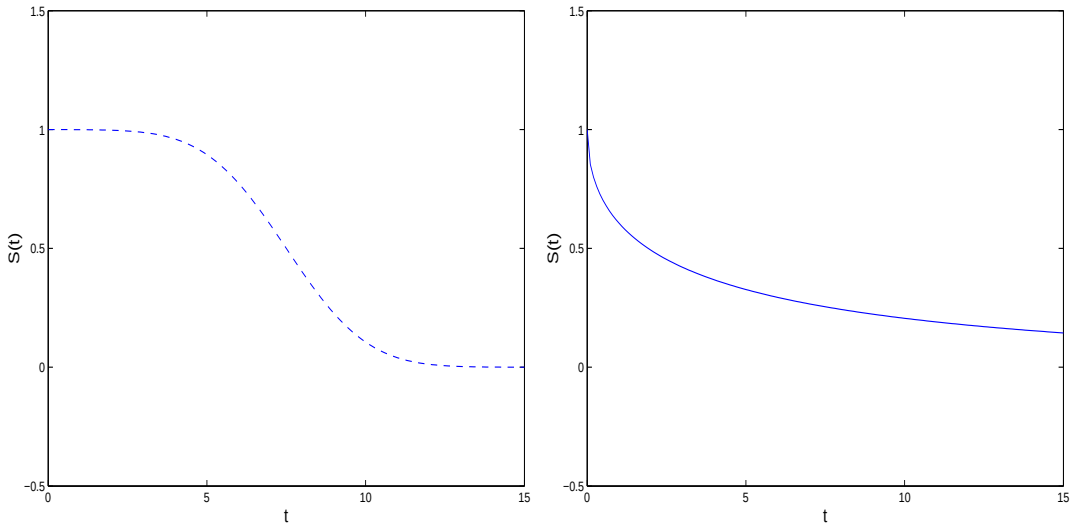
Ettersom overlevelsesfunksjonen er komplementet til den kumulative tetthetsfunksjonen blir overlevelsesfunksjonen arealet under sannsynlighetstetthetsfunksjonen fra tid t og oppover

$$S(t) = \int_t^\infty f(u)\mathrm{d}u.$$

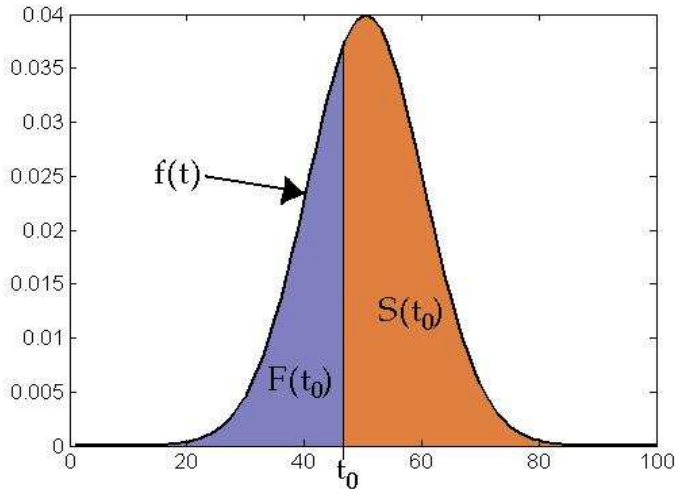
Til sammen utgjør $S(t)$ og $F(t)$ hele arealet under tetthetsfunksjonen og summerer derfor opp til 1. Sammenhengen mellom integrasjon og derivasjon gir oss videre at $f(t) = -\frac{\partial S(t)}{\partial t}$. Se figur 3.5 for en grafisk framstilling av sammenhengen mellom $S(t)$, $F(t)$ og $f(t)$.

Sammenhengen mellom hasardraten, fordelingsfunksjonen og overlevelsesfunksjonen kan skrives som

$$h(t) = \frac{f(t)}{S(t)}.$$



Figur 3.4: Til venstre sees overlevelsesfunksjonen til en normalfordeling med parametere $\mu = 7.5$ og $\sigma = 2$ og til høyre overlevelsesfunksjonen til en Weibullfordeling med parametere $\kappa = 0.5$ og $\tau = 0.5$. Disse tilsvarer tetthetsfunksjonene i figur 3.1 og hasardfunksjonene i figur 3.2



Figur 3.5: Illustrasjon av sammenhengen mellom $f(t)$, $S(t)$ og $F(t)$. $f(t)$ er den svarte kurven, $F(t_0)$ er arealet under denne kurven opp til tid t_0 og $S(t_0)$ er arealet under kurven fra t_0 og oppover.

Denne sammenhengen kan vises ved å stokke litt om på uttrykket for hasardraten

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < (t + \Delta t) \mid T \geq t)}{\Delta t} = \lim_{\Delta t \to 0} \frac{P(t \leq T < (t + \Delta t))}{\Delta t \; P(T \geq t)} = f(t) \frac{1}{S(t)}.$$

Man kan anta at overlevelsesfunksjonen følger ulike fordelinger. Enhver fordeling over ikke-negative verdier er en potensiell kandidat. Kandidater til kontinuerlige fordelinger er for eksempel

- Weibullfordelingen
- Eksponentialfordelingen
- Normalfordelingen
- Log-normalfordelingen.

Valg av fordeling bør foretas på bakgrunn av flere kriteriter. I første rekke bør man naturligvis velge en fordeling som samsvarer godt med den virkelige fordelingen av observasjonene. Ut over dette er det å kunne sette opp eksplisitte uttrykk for likelihoodfunksjonen for en valgt modell i ulike situasjoner klart en fordel. I tillegg bør man vurdere eksistens og enkelhet av $S(t)$, $f(t)$ og $h(t)$. Man bør kunne sette opp uttrykk for disse funksjonene for alle verdier av t, og uttrykkene for dem bør helst være forholdsvis enkle. I tabell 3.2 kan man se ulike egenskaper ved de parametriske fordelingene som ble nevnt her. Ut fra tabellen kan man sammenlikne uttrykkene for de ulike fordelingene, og den kan brukes som en referanse i senere kapitler.

3.3 Modelltilpasning

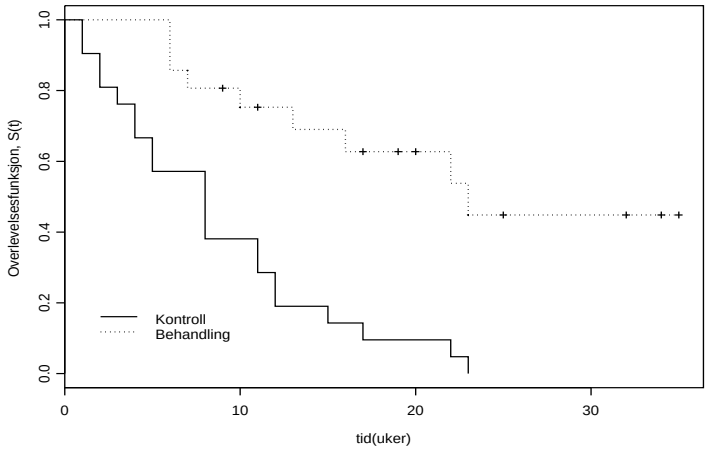
Ved å gjøre en regresjon kan man tilpasse en modell ved hjelp av data man har samlet inn. En slik modelltilpasning vil resultere i estimater for et eller flere av begrepene i avsnittet over. En slik tilpasning kan gjøres enten parametrisk eller ikke-parametrisk.

I en *parametrisk modelltilpasning* antar man at funksjonen som skal estimeres bestemmes helt av en vektor med parametere $\boldsymbol{v}$, det vil si at funksjonen har form som en parametrisert mapping $f(\cdot|\boldsymbol{v})$. I vår situasjon vil dette si at vi antar at overlevelsesfunksjonen karakteriserer en gitt fordeling som kan beskrives ved en eller flere parametere. Dette kan gi enkle analyser, ettersom man kan benytte seg av standard ML-teori (se appendiks A) for å estimere disse parameterne. Resultatene av en slik modelltilpasning blir også enkle å presentere, ettersom man bare trenger å oppgi fordelingen og de estimerte parameterne[25]. En annen fordel med modeller estimert på denne måten er at man kan oppnå høy presisjon på prediksjon av overlevelsen til nye observasjoner, og høy presisjon på eventuelle hypotesetester[28]. En populær parametrisk modell for overlevelsesdata vil bli presentert i avsnitt 3.5.

Alternativet er å bruke det som kalles for *ikke-parametrisk modelltilpasning*. Da antar man mindre om funksjonen $f(\cdot)$ som skal estimeres. Slike tilpasningsmetoder er ofte nyttige når formen til den sanne regresjonsflaten er ukjent. I vårt tilfelle vil dette si at man ikke antar at overlevelsesfunksjonen karakteriserer en gitt fordeling. Man gjør dermed færre forutsetninger

Fordelinger:	**Eksponential**	**Weibull**	**Normal**	**Log-normal**
Parametere	$\lambda > 0$, skaleringsparameter	$\tau > 0$, formparameter $\kappa > 0$, skaleringsparameter	$\sigma > 0$, variansparameter μ, lokaliseringsparameter	$\sigma > 0$, variansparameter μ, lokaliseringsparameter
Forventning	$\kappa^{-1/\tau}\frac{1}{\lambda}$	$\Gamma(1+1/\tau)$	μ	$exp(\mu + 0.5\sigma^2)$
Varians	$\frac{1}{\lambda^2}$	$\kappa^{-2/\tau}\{\Gamma(1+\frac{2}{\tau}) - [\Gamma(1+\frac{1}{\tau})]^2\}$	σ^2	$e^{2\mu}e^{\sigma^2}\left[e^{\sigma^2}-1\right]$
Hasardrate $h(t)$	λ	$\tau\kappa t^{\tau-1}$	$\frac{f(t)}{S(t)}$	$\frac{f(t)}{S(t)}$
Overlevelsesfunksjon $S(t)$	$exp(-\lambda t)$	$exp(-\kappa t^{\tau})$	$1-\Phi[\frac{t-\mu}{\sigma}]$	$1-\Phi[\frac{ln(t)-\mu}{\sigma}]$
Sannsynlighetsfunksjon $f(t)$	$\lambda\ exp(-\lambda t)$	$\tau\kappa t^{\tau-1}exp(-\kappa t^{\tau})$	$\frac{exp\left[-\frac{1}{2}(\frac{t-\mu}{\sigma})^2\right]}{(2\pi)^{1/2}\sigma}$	$\frac{exp\left[-\frac{1}{2}(\frac{ln(t)-\mu}{\sigma})^2\right]}{t(2\pi)^{1/2}\sigma}$

Tabell 3.2: Hasardrate, overlevelsesfunksjon og sannsynlighetsfordelingsfunksjon for noen vanlige overlevelsesfordelinger. Φ er den kumulative fordelingsfunksjonen for en standard normalfordelt variabel og Γ er gammafunksjonen.



Figur 3.6: Kaplan-Meier kurver for to pasientgrupper, der en gruppe har fått behandling mens den andre er en kontrollgruppe. Ved å utføre tester for disse estimatene kan man si noe om pasienter i behandlingsgruppen har større sjanse for å overleve enn pasienter i kontrollgruppen.

enn de parametriske metodene. Dersom man valgte en parametrisk modell der antagelsene man gjorde for modellen ikke var korrekte, vil dette kunne gi feilaktige svar. Ikke-parametriske metoder kan også gi mer robuste estimater enn om man velger en parametrisk modell der forutsetningene som ble gjort for modellen var gale. Et eksempel på en slik estimator er Kaplan-Meier estimatoren, som vil bli presentert nærmere i neste avsnitt.

3.4 Kaplan-Meier estimatoren

Kaplan-Meier estimatoren er et eksempel på en ikke-parametrisk estimator for overlevelsesfunksjonen. La Y_i være antall individer under risiko ved tid t_i, altså antall individer som like før tid t_i verken har opplevd hendelse eller har blitt sensurert. Anta videre at datasettet består av D ulike usensurerte hendelsestider $t_{(1)} < \ldots < t_{(D)}$, og at d_i er antall personer som dør ved tid t_i. Da er $\frac{d_i}{Y_i}$ et estimat for den betingede sannsynligheten for at et individ som lever fram til t_i vil oppleve hendelsen ved t_i. Kaplan-Meier estimatoren er definert som følger for alle verdier av t i intervallet $[0, t_{max}]$, der t_{max} er den største observerte verdien man har for t:

$$\hat{S}(t) = \begin{cases} 1 & \text{hvis } t < t_1 \\ \prod_{t_i \leq t}[1 - \frac{d_i}{Y_i}] & \text{hvis } t_1 \geq t \end{cases}.$$

Denne estimatoren er ikke definert for verdier av t som er større enn den største observasjonen man har for t. Estimatoren blir en step-funksjon som hopper for hver gang en hendelse inntreffer. Størrelsen på hopp i bestemmes av antall hendelser ved tid t_i, og mønsteret av sensurerte observasjoner før t_i. Se figur 3.6 for eksempler på Kaplan-Meier kurver. Ettersom antall observasjoner blir større vil Kaplan-Meier estimatet konvergere mot den sanne overlevelsesfunksjonen[28].

Kaplan-Meier estimatoren er en meget populær estimator. Årsaker til dette er blant annet at den har en intuitiv fortolkning, i tillegg til at den er lett å estimere. Et viktig poeng er at det også finnes metoder for å måle usikkerheten til estimatoren, slik at det er mulig å kunne si noe om forskjellen i overlevelse mellom grupper. Blant annet er *log-rank testen* en mye brukt test for denne estimatoren [28]. Innenfor denne rammen blir det lett å teste om for eksempel overlevelsen for en behandling er bedre enn for en annen behandling. Videre har Kaplan-Meier estimatoren den fordelen at den gir en meget enkel grafisk framstilling av overlevelsesdata som er forståelig, selv uten inngående kjennskap til statistisk overlevelsesanalyse. Alle disse egenskapene bidrar til at estimatoren er mye brukt, særlig innen biologi og medisin.

3.5 Weibullfordelingen

Weibullfordelingen er en mye anvendt fordeling innen overlevelsesanalyse. Den tillater flere ulike former på hasarden, noe som gir en svært fleksibel parametrisk modell. I tillegg gir den enkle uttrykk for hasard- og overlevelsesfunksjonen (se tabell 3.2). Fordelingen er bestemt av to parametere, $\kappa > 0$ og $\tau > 0$, der κ er en skaleringsparameter, mens τ bestemmer hvilken form hasarden får. Hasardraten for Weibullfordelingen er gitt ved

$$h(t) = \tau\kappa t^{\tau-1}.$$

Hvis $\tau > 1$ blir hasarden en økende funksjon av tiden, og hvis $\tau < 1$ så blir hasarden en avtagende funksjon av tiden. Eksponentialfordelingen er et spesialtilfelle av Weibullfordelingen når $\tau = 1$. Overlevelsesfunksjonen for Weibullfordelingen er gitt ved

$$S(t) = exp(-\int_0^t h(u)\mathrm{d}u) = exp(-\int_0^t \tau\kappa u^{\tau-1}\mathrm{d}u) = exp(-\kappa t^{\tau}).$$

Dersom man redefinerer parameterne til Weibullfordelingen med $\mu = -log(\kappa)\sigma$ og $\sigma = 1/\tau$ så får man sammenhengen[28]:

$$log\ T = \mu + \sigma W, \tag{3.2}$$

hvor W følger standard ekstremverdifordelingen med sannsynlighetstetthetsfunksjon $f_W(w) = exp(w - exp(w))$. Som vi skal se senere kan det være nyttig å kunne uttrykke Weibullfordelingen både ved hjelp av denne log-lineære modellen og på "vanlig" måte med hasarden og overlevelsesfunksjonen.

3.6 Regresjonsmodeller

Fram til nå har vi sett på modellering av overlevelsestider i en situasjon der disse antas å være identisk fordelte, slik at man kan tenke på dem som ulike realiseringer av den samme stokastiske variabelen T. I praksis ønsker man ofte å inkorporere forklaringsvariable (kovariater) med individspesifikke verdier i modellen, slik at fordelingen til den enkelte levetid avhenger av individets forklaringsvariable. I dette tilfellet må man tenke på levetidene som realiseringer av ulike stokastiske variable $T_1, \ldots, T_n$. Man ønsker fortsatt at disse følger samme type fordeling (for eksempel Weibull), men parameterne i fordelingen vil kunne avhenge av kovariatene.

Anta at vi for hvert individ i har en vektor $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^T$ med kovariater. Eksempler på kovariater kan være kvantitative variable som blodtrykk, kroppstemperatur, alder og vekt, eller kvalitative variable som kjønn, behandling og sykdomsstatus. I denne oppgaven vil $\mathbf{x}_i$ inneholde genekspresjonsdata fra p gener. En regresjon går ut på å modellere sammenhengen mellom en eller flere av disse kovariatene og responsen (altså levetiden), på bakgrunn av et gitt datasett. Ved å gjøre dette kan man for eksempel sammenlikne effekten av ulike behandlinger eller si noe om sammenhengen mellom ulike kovariater og overlevelsen til en pasient. Etter en slik tilpasning av dataene ender man opp med en estimert modell som kan brukes til prediksjon av overlevelse for nye individer, eller man kan gå videre og studere disse kovariatene for å få større innsikt i mekanismene som gjør at nettopp disse er signifikante. Statistiske strategier for regresjon med overlevelsesdata likner de fra vanlig lineær regresjon, men detaljene i regresjonsteknikkene for overlevelsesanalyse er ulike.

Det er to klassiske måter å modellere effekten av kovariatene på overlevelsestiden. Den første er analog med ordinær lineær regresjon og går under navnet akselererte levetidsmodeller. Logaritmen til overlevelsesfunksjonen modelleres da som en respons av en lineær kombinasjon av kovariatene. I den andre metoden modelleres effekten av kovariatene direkte på en funksjon som beskriver fordelingen til T. Vanligvis gjøres dette på hasardfunksjonen, men dette vil igjen også påvirke de andre funksjonene, som overlevelsesfunksjonen og fordelingsfunksjonen. Resultatet blir modeller som er betinget på kovariatene $\mathbf{x}$, for eksempel $h(t|\mathbf{x})$ og $S(t|\mathbf{x})$. En vanlig måte å gjøre dette på er ved å bruke proporsjonale hasardmodeller.

Vi starter med å se på de akselererte levetidsmodellene. Vi skal se at Weibullfordelingen kan uttrykkes ved en slik modell, og at denne også kan uttrykkes ved en proporsjonal hasardmodell. Videre skal vi se nærmere på proporsjonale hasardmodeller og introdusere *Cox proporsjonale hasardmodell*, som er et av de mest populære alternativene innenfor slike modeller. Til slutt skal vi se på mulighetene for prediksjon av overlevelse, og litt om problemer rundt dette.

3.7 Akselererte levetidsmodeller

Akselererte levetidsmodeller antar et lineært forhold mellom kovariatene og logaritmen til overlevelsestiden. Et slikt forhold kan skrives som en lineær modell på formen

$$log\ T = \gamma_0 + \boldsymbol{\gamma}^T \mathbf{x} + \sigma\epsilon, \tag{3.3}$$

der $\boldsymbol{\gamma}^T = (\gamma_1, \ldots, \gamma_p)$ er vektoren av regresjonsparametere, og ϵ er støyleddet som følger feilfordelingen som er skalert med en positiv konstant σ. Ulike valg av denne feilfordelingen gir ulike regresjonsmodeller. Hvis for eksempel ϵ følger ekstremverdifordelingen, gir (3.3) Weibullfordelte levetider T. Hvis ϵ følger normalfordelingen, gir (3.3) log-normalfordelte levetider T. Ettersom tiden T alltid er positiv tar man logaritmen av tiden på venstresiden. Dette gir en transformert respons som tilsvarer verdiområdet til høyresiden i modellen. Slike modeller introduserer effekten av kovariatene i overlevelsesfunksjonen, hasardfunksjonen og andre funksjoner som beskriver fordelingen til T. Vi skal se litt nærmere på hvordan denne lineære effekten blir på overlevelsesfunksjonen. Vi setter inn uttrykket for T fra (3.3) i den betingede

overlevelsesfunksjonen $S(t|\mathbf{x})$ ved

$$\begin{aligned} S(t|\mathbf{x}) = P(T > t \mid \mathbf{x}) &= P(log\ T > log\ t \mid \mathbf{x}) \\ &= P(\gamma_0 + \boldsymbol{\gamma}^T\mathbf{x} + \sigma\epsilon > log\ t \mid \mathbf{x}) \\ &= P(\gamma_0 + \sigma\epsilon > log\ t - \boldsymbol{\gamma}^T\mathbf{x} \mid \mathbf{x}) \\ &= P(e^{\gamma_0+\sigma\epsilon} > te^{-\boldsymbol{\gamma}^T\mathbf{x}} \mid \mathbf{x}). \end{aligned}$$

La $S_0(t)$ være overlevelsesfunksjonen når alle kovariatene er null, det vil si når (3.3) degenererer til $T = exp\{\gamma_0+\sigma\epsilon\}$. Da er $S_0(t) = P(exp\{\gamma_0+\sigma\epsilon\} > t)$. Fordelingen til denne basisfunksjonen bestemmes da parametrisk etter hvilken fordeling man antar for støyleddet ϵ, på samme måte som beskrevet over. (Den er altså parametrisk bestemt av feilfordelingen.) Dersom man velger å uttrykke $S(t|\mathbf{x})$ ved hjelp av $S_0(t)$, kan uttrykket over skrives som

$$S(t|\mathbf{x}) = S_0(te^{-\boldsymbol{\gamma}^T\mathbf{x}}|\mathbf{x}).$$

En slik modell kalles for en *akselerert levetidsmodell*, ettersom den lineære modellen for $log\ T$ leder til en skalering av overlevelsestiden. Dette kommer av at faktoren $exp(\boldsymbol{\gamma}^T\mathbf{x})$ blir en akselerasjonsfaktor på levetiden, som sier hvor mye endringer i kovariatverdiene vil endre overlevelsen fra basisen, det vil si dersom man hadde satt alle kovariatene lik 0. Akselererte levetidsmodeller kan sees på som et spesialtilfelle av lineære modeller, men de begrenses ved hvilke fordelinger man kan anta for støyleddet W. En slik modell er en parametrisk modell, ettersom man gjør en antagelse om at basisoverlevelsesfunksjonen, $S_0(t)$, følger en gitt fordeling.

Et alternativ til akselererte levetidsmodeller er det som kalles proporsjonale hasardmodeller, eller multiplikative hasardmodeller, som modellerer hasardfunksjonen som en funksjon av kovariatene. Fordelen med dette er at man kan tilpasse en modell uten å måtte gjøre noen antagelser om fordelingen til hasardfunksjonen. Før vi skal se nærmere på proporsjonale hasardmodeller vil ulike modeller for Weibullfordelingen bli presentert. Denne fordelingen er den eneste som har den egenskapen at den kan uttrykkes både som en akselerert levetidsmodell og som en proporsjonal hasardmodell.

3.8 Simulering av levetider fra en Weibullfordeling

For å kunne simulere levetidsdata fra en Weibullfordeling, slik at levetidene avhenger av en mengde kovariater må man ha en metode for å introdusere kovariater i fordelingen. På bakgrunn av slike data kan man senere tilpasse en utvalgt modell. Weibullfordelingen har den fordelen at den kan representeres både som en akselerert levetidsmodell og en proporsjonal hasardmodell på formen

$$h(t|\mathbf{x}) = h_0(t)e^{\boldsymbol{\beta}^T\mathbf{x}}. \tag{3.4}$$

Vi skal se nærmere på slike modeller i neste avsnitt. Denne egenskapen ved Weibullfordelingen gir mulighet til å simulere data ved hjelp av den enkle lineære sammenhengen i en akselerert levetidsmodell, for så å tilpasse en proporsjonal hasardmodell på bakgrunn av disse dataene. For å kunne gjøre dette må man finne en sammenheng mellom parameterne i en akselerert levetidsmodell, som i (3.3), parameterne i Weibullfordelingen, τ og κ, og parameterne til den proporsjonale hasardmodellen, $\boldsymbol{\beta}$. Dersom vi definerer $\sigma = 1/\tau$, $\gamma_0 = -log(\kappa)/\tau$ og

$\gamma_j = -\beta_j/\tau$ for $j = 1, \ldots, p$, og setter dette inn i uttrykket for den akselererte levetidsmodellen (3.3) får vi,

$$log\ T = \frac{1}{\tau}(-log\ \kappa - \boldsymbol{\beta}^T\mathbf{x} + W), \tag{3.5}$$

der W følger standard ekstremverdifordelingen. Sammenhengen i (3.5) kan dermed brukes til å simulere levetider fra Weibullfordelingen. Dette kan man gjøre ved å generere kovariatvektorer $\mathbf{x} \in \mathbb{R}^p$ og velge en $\boldsymbol{\beta} \in \mathbb{R}^p$ til å være den sanne regresjonsparameteren. Man kan for eksempel sette $\beta_1, \ldots, \beta_r = s$, der $s \neq 0$ bestemmer at de r første kovariatene skal ha betydning for overlevelsestiden. Resten av koeffisientene settes lik 0, $\beta_{r+1}, \ldots, \beta_p = 0$, noe som medfører at de $p - r$ siste kovariatene ikke har noen innvirkning på overlevelsestiden. Man velger så verdier for parameterne κ og τ og setter alle disse verdiene inn i uttrykket (3.5). For hver levetid trekkes et støyledd W_i fra ekstremverdifordelingen. Dette kan gjøres ved å bruke at $W = -log(-log(R))$ er ekstremverdifordelt når $R \sim U(0,1)$[44]. Dette resulterer i simulerte levetider $\tilde{t}_i$ for $i = 1, \ldots, n$. For å innføre sensurering av levetidene kan man definere sensurerte levetider som

$$t_i = min(\tilde{t}_i, t_c),$$

der t_c er en kuttetid, det vil si høyeste mulige observerte levetid. Denne kan for eksempel velges slik at en ønsket andel, c, av levetidene blir sensurert. Sensurindikatorene δ_i settes så ved:

$$\delta_i = \begin{cases} 0 & \text{hvis } \tilde{t}_i > t_c \\ 1 & \text{hvis } \tilde{t}_i \leq t_c \end{cases}.$$

Vektoren $\boldsymbol{\beta}$ er da den sanne koeffisientvektoren som man ønsker å estimere igjen. Ettersom data er simulert fra Weibullfordelingen vil man kunne estimere denne parametervektoren ved å tilpasse dataene til en proporsjonal hasardmodell, lik den vi så i (3.4). Dette vil bli utnyttet senere i oppgaven. For oversiktens skyld er hele prosedyren for å simulere Weibull-fordelte levetider med kovariater gitt nedenfor:

1. Kovariatverdier, $\mathbf{x}_i$ for $i = 1, \ldots, n$, kan simuleres dersom dette er nødvendig. Dette kan for eksempel gjøres ved å trekke korvariatverdier fra en passende fordeling, og i tillegg kontrollere korrelasjonen mellom genekspresjonsvektorene. Alternativt kan man bruke eksisterende kovariatverdier og simulere levetider for disse.

2. Bestem antall kovariater, r, som skal ha betydning for levetidene, og hvor stor betydning de skal ha. Dette resulterer i den sanne parametervektoren $\boldsymbol{\beta}$.

3. Bestem parameterne til Weibullfordelingen, $\kappa > 0$ og $\tau > 0$.

4. For $i = 1, \ldots, n$ beregn usensurerte levetider $\tilde{t}_i$ etter uttrykket i (3.5), ved å simulere støy W_i fra ekstremverdifordelingen.

5. Bestem kuttetid t_c slik at en ønsket andel c blir sensurerte, og finn sensurerte levetider, t_i og sensurstatus, δ_i.

3.8.1 Parametrisk versus ikke-parametrisk tilpasning av data

I denne oppgaven vil altså data bli simulert fra en akselerert levetidsmodell, men tilpasset med en proporsjonal hasardmodell. Det å tilpasse dataene til en akselerert levetidsmodell tilsvarer

å gjøre en parametrisk tilpasning. En akselerert levetidsmodell har formen

$$log\ T \sim \boldsymbol{\beta}^T \mathbf{x} + \sigma\epsilon, \tag{3.6}$$

der σ er en skaleringsparameter for støylesset og fordelingen til støyleddet ϵ bestemmer fordelingen til T. Dersom man definerer at ϵ skal være ekstremverdifordelt tilsvarer dette den akselererte levetidsmodellen for Weibullfordelingen (3.2), det vil si den Weibullmodellen som vil bli brukt til simulering av data i denne oppgaven. Det å tilpasse data til en Cox proporsjonale hasardmodell vil tilsvare å gjøre en ikke-parametrisk tilpasning av data. Ettersom Weibullfordelingen kan representeres både ved en akselerert levetidsmodell og en proporsjonal hasardmodell er det mulig å simulere data fra den ene modellen og tilpasse dem med den andre. Parameterestimatene blir ikke nødvendigvis de samme for begge modeller. I et arbeid av Solomon[47] vises det at i en analyse der man antar en proporsjonal hasardmodell når den underliggende fordelingen for hendelsestidene egentlig er en akselerert levetidsmodell, vil den sanne regresjonsvektoren være komponentvis proporsjonal til grensen av den sanne fordelingen for ML estimatet. Det vil si at dersom vi definerer $\boldsymbol{\gamma}$ som den sanne parametervektoren for den akselererte levetidsmodellen og $\hat{\boldsymbol{\beta}}$ som ML-estimatet for den proporsjonale hasardmodellen, så er $\hat{\boldsymbol{\beta}}_j = a\gamma_j$ for $j = 1, \ldots, p$. Dette medfører at dersom man antar en proporsjonal hasardmodell under analysen når en akselerert levetidsmodell egentlig er den korrekte modellen, så vil altså den relative effekten av forklaringsvariablene være uendret. Det er derfor mulig å estimere en Cox-modell på bakgrunn av dataene som genereres i denne oppgaven, til tross for at disse genereres fra en akselerert levetidsmodell. Ved å bruke en proporsjonal hasardmodell vil kunne plukke ut de samme relevante kovariatene.

3.9 Proporsjonale hasardmodeller

Den andre tilnærmingen til å inkludere kovariateffekter i regresjonsmodellen er å modellere hasarden som en funksjon av kovariatene. En populær måte å gjøre dette på er ved det som kalles *proporsjonale hasardmodeller*. Her er den betingede hasarden $h(t|\mathbf{x})$ definert som et produkt av en baselinehasard $h_0(t)$ som er lik for alle individer og en ikke-negativ linkfunksjon av kovariatene $c(\boldsymbol{\beta}^T\mathbf{x})$, det vil si

$$h(t|\mathbf{x}) = h_0(t)c(\boldsymbol{\beta}^T\mathbf{x}). \tag{3.7}$$

Generelt kan enhver ikke-negativ funksjon brukes som $c(\cdot)$. Kovariatene virker multiplikativt på baselinehasarden, det vil si de øker eller minsker hasarden relativt til $h_0(t)$. Et særtrekk ved multiplikative hasardmodeller er at hasardratioen til to personer med ulike verdier for kovariatene $\mathbf{x}$ vil være proporsjonal dersom kovariatene ikke er tidsavhengige. Dersom man skriver ut ratioen for hasardene til person i og j, ser man at det blir en konstant som er uavhengig av tiden:

$$\frac{h(t|\mathbf{x}_i)}{h(t|\mathbf{x}_j)} = \frac{h_0(t)c(\boldsymbol{\beta}^T\mathbf{x}_i)}{h_0(t)c(\boldsymbol{\beta}^T\mathbf{x}_j)} = \frac{c(\boldsymbol{\beta}^T\mathbf{x}_i)}{c(\boldsymbol{\beta}^T\mathbf{x}_j)} = \eta_{ij}.$$

Det kan være flere grunner til at man ønsker å gjøre en slik antagelse om proporsjonalitet. En grunn er at det gjør det lettere å regne på og analysere modellene. Videre blir det enkelt å tolke de estimerte parameterne. Anta at link-funksjonen $c(\boldsymbol{\beta}^T\mathbf{x})$ er på formen $exp(\boldsymbol{\beta}^T\mathbf{x})$ og at to personer har ulike verdier for x_1, slik at den ene personen har $x_1 = 1$ som for eksempel

indikerer behandling og den andre personen har $x_1 = 0$ som indikerer en placebobehandling. Dersom alle de andre kovariatene x_i er like for begge personene er hasardratioen for de to personene lik $exp(\beta_1)$. Denne kan da tolkes som relativ risiko for at en person opplever en hendelse dersom vedkommende fikk behandling, i forhold til om vedkommende ikke hadde fått behandling. Proporsjonale hasarder er i mange tilfeller en helt gyldig antagelse, men man bør teste om antagelsen er riktig i hvert enkelt tilfelle. Det finnes blant annet en del grafiske tester for å sjekke om denne antagelsen er rimelig [28, 3]. Vi skal se nærmere på en av de mest populære måtene å tilpasse en slik modell på.

3.10 Cox proporsjonale hasardmodell

En meget populær framgangsmåte for estimering av slike modeller ble foreslått av Cox i 1972[8], og går derfor ofte under navnet *Cox proporsjonale hasardmodell*. (Denne vil bli referert til som *Cox- modellen* videre i oppgaven.) Funksjonen $c(\cdot)$ fra (3.7) vil i denne modellen ha formen $exp\{\boldsymbol{\beta}^T\mathbf{x}\}$ og den betingede hasardraten blir da

$$h(t|\mathbf{x}) = h_0(t)exp\{\boldsymbol{\beta}^T\mathbf{x}\}. \tag{3.8}$$

For å finne et estimat for $\boldsymbol{\beta}$ i denne modellen kan man benytte seg av maksimum likelihood (ML) metoden. Vi starter derfor med å definere likelihooden til denne modellen. Problemet er at denne fulle likelihooden vil bli en funksjon som avhenger både av $\boldsymbol{\beta}$ og baselinehasarden $h_0(t)$. Det er vanlig i denne modellen å bruke det som kalles en partiell likelihood istedenfor den fulle likelihooden. Bakgrunnen for dette er at det man som regel ønsker å gjøre er å estimere regresjonsvektoren $\boldsymbol{\beta}$. Ut fra dette blir den generelle hasardfunksjonen, $h_0(t)$ fra (3.7), bare en forstyrrende parameter som man derfor vil prøve å fjerne fra likelihooden. En måte å gjøre dette på er å finne et uttrykk for $h_0(t)$ ved å maksimere den fulle likelihooden med hensyn på $h_0(t)$ for en fast $\boldsymbol{\beta}$. Dette uttrykket setter man så inn for $h_0(t)$ i den fulle likelihooden, og resultatet er det som kalles en partiell likelihood som bare avhenger av $\boldsymbol{\beta}$. Denne tilnærmingen til å fjerne en forstyrrende parameter kalles for *profilering*. Asymptotisk gir den partielle likelihooden identiske resultater som den fulle likelihooden, og det er derfor fullt mulig å gjøre inferens rundt $\hat{\boldsymbol{\beta}}$ [37]. I tillegg har den partielle likelihooden en del andre fordeler. Blant annet får man et mye enklere uttrykk for likelihooden å jobbe med, i tillegg til at man ikke alltid trenger å estimere $h_0(t)$. Dersom man ønsker det kan denne estimeres ikke-parametrisk senere. Modellen omtales ofte som *semiparametrisk*, ettersom parametrisk form bare antas for kovariateffekten, mens baselinehasarden behandles ikke-parametrisk.

Anta at vi har overlevelsesdata for n individer, der alle hendelsestidene er forskjellige. La $t_1 < \ldots < t_n$ være de ordnede observerte hendelsestidene, $\delta_1, \ldots, \delta_n$ er tilhørende sensurstatus som antas å være *ikke-informativ*. Dette betyr at gitt kovariatene er det uavhengighet mellom utfallet av hendelsen og tiden for hendelsen til en pasient. Kovariatvektorene som assosieres med de ulike individene er $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Da blir den partielle likelihooden for Cox modellen

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \left\{ \frac{exp(\boldsymbol{\beta}^T\mathbf{x}_i)}{\sum_{j\in\mathcal{R}_{(t_i)}} exp(\boldsymbol{\beta}^T\mathbf{x}_j)} \right\}^{\delta_i}. \tag{3.9}$$

Her vil $\mathcal{R}_{(t_i)} = \{j : t_j \geq t_i\}$ angi risikomengden ved tid t_i, det vil si alle de individer som fremdeles var i live og usensurerte rett før dette tidspunktet. Legg merke til at bare hendel-

sestider bidrar til telleren i den partielle likelihooden, mens både sensurerte og usensurerte individer tas med i nevneren i brøken, da risikomengden inkluderer alle de som er i live ved tidspunktet. Denne partielle likelihooden (3.9) kan behandles som en vanlig likelihood, og maksimeres med hensyn på $\boldsymbol{\beta}$ ved bruk av vanlige ML-metoder. Dette blir ekvivalent med å maksimere logaritmen til denne partielle likelihooden, som er

$$log\ L(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) = \sum_{i=1}^{n} \delta_i(\boldsymbol{\beta}^T \mathbf{x}_i) - \sum_{i=1}^{n} \delta_i log \left\{ \sum_{j \in \mathcal{R}_{(t_i)}} exp(\boldsymbol{\beta}^T \mathbf{x}_j) \right\}. \tag{3.10}$$

Den partielle log-likelihooden kan maksimeres med en iterativ optimeringsalgoritme. Man partiellderiverer da (3.10) med hensyn på komponentene i $\boldsymbol{\beta}$, og løser så de p ulineære likningene $\frac{\partial}{\partial \beta_q} l(\boldsymbol{\beta}) = 0$ for $q = 1, \ldots, p$, noe som kan gjøres numerisk. Legg merke til at dette ikke kan løses eksakt [3]. Resultatet av optimeringen blir altså en $\hat{\boldsymbol{\beta}}$ som maksimerer den partielle log-likelihooden. Slike optimeringsalgoritmer vil vi komme tilbake til i kapittel 5.

3.11 Samtidige hendelser

Problemer oppstår ved bruk av Cox-modellen når flere hendelser inntreffer på samme tidspunkt. Fra (3.9) ser man at modellen bygger på at det er en en-til-en sammenheng mellom hendelsestider og individer, i og med at man må vite hvilke som faktisk er under risiko da en person opplever en hendelse. Å være under risiko vil si at man verken har opplevd hendelse eller har blitt sensurert før tidspunktet da hendelsen inntreffer. En slik en-til-en sammenheng mellom hendelsestidspunkter er ofte ikke tilfellet da dødstidspunkt ikke alltid måles kontinuerlig, men diskret (for eksempel i dager eller uker), alt ettersom hvordan undersøkelsen er lagt opp. Dette gjør at det blir mulig for to personer å dø "samtidig", og man må finne en metode for å håndtere slike samtidige hendelser.

En enkel måte å løse problemet på er å si at alle de som har hendelser på samme tidspunkt fremdeles er i risikomengden ved det tidspunktet. Dette innebærer at individene med samtidige hendelser får samme risikomengde. Denne metoden er oppkalt etter Breslow som innførte den i 1974[7]. Det finnes også andre, mer sofistikerte metoder for å håndtere slike samtidige hendelser, men de vil ikke bli presentert nærmere her [28, 9, 25]. Likelihooden ved Breslow-håndtering av samtidige hendelser kan defineres som følger. La $t_1 < t_2 < \ldots < t_D$ være de D ulike ordnede usensurerte hendelsestidene. La videre d_i være antall hendelser ved tid t_i og $\mathbb{D}_i$ være mengden av alle individer som dør ved tid t_i. La $\mathbf{z}_i = \sum_{j \in \mathbb{D}_i} \mathbf{x}_i$, og $\mathcal{R}_{(t_i)}$ er mengden av individer under risiko rett før t_i. Under Breslows tilnærming får vi da følgende uttrykk for den partielle likelihooden

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{D} \frac{exp(\boldsymbol{\beta}^T \mathbf{z}_i)}{[\sum_{j \in \mathcal{R}_{(t_i)}} exp(\boldsymbol{\beta}^T \mathbf{x}_j)]^{d_i}} \tag{3.11}$$

I likelihooden anses altså de d_i hendelsene på et gitt tidspunkt som forskjellige. Deres bidrag til likelihoodfunksjonen beregnes, og man finner bidraget til likelihooden ved å multiplisere over alle hendelser ved tid t_i. Når det er få samtidige hendelser fungerer denne approksimasjonen rimelig bra[28].

3.12 Prediksjon av overlevelse

For en gitt pasient vil det ofte kunne være ønskelig å gi et estimat for hvor lenge vedkommende har igjen å leve. Ved å tilpasse en Cox-modell på bakgrunn av genekspresjonsdata ønsker man å kunne bruke denne til å stille slike prognoser ut fra målinger av en pasients genekspresjonmønster. Et mål under tilpasning av modellen er altså å finne den modellen som best klarer å predikere overlevelsestider for nye individer. Et viktig prinsipp i denne sammenhengen er at enhver prediksjonsmodell optimalt sett bør valideres ved å teste den ut på uavhengige pasientdata for å kunne angi hvor godt modellen fungerer på alle data, ikke bare de som ble brukt under estimeringen av modellen.

Det å bruke Cox-modellen (3.8) direkte for å predikere overlevelsestider viser seg derimot problematisk. Slike estimater for overlevelse fra denne modellen får ofte svært høy varians, og vil dermed gi meget usikre estimater for levetiden. En annen kompliserende faktor er at man ikke får noe estimat for hasarden utenfor de observerte levetidene i datasettet som modellen estimeres på bakgrunn av, så det er vanskelig å bygge en prediktiv modell for alle mulige levetider.

Vi skal se på en mulig tilnærming til prediksjon av levetider der man unngår disse problemene. Dersom man tar utgangspunkt i uttrykket for den betingede hasarden for Cox-modellen (3.8) eller den partielle likelihooden til Cox-modellen (3.9) ser man at kovariatene kun inngår i denne modellen gjennom en lineær funksjon på formen:

$$f(\mathbf{x}) = \beta_1 x_1 + \ldots + \beta_p x_p = \boldsymbol{\beta}^T \mathbf{x}.$$

Denne lineære kombinasjonen kalles ofte for *prognostisk indeks* eller *risikoindeks* i denne sammenhengen. Man ser at hvert individs kovariater påvirker levetiden kun gjennom denne lineære kombinasjonen av kovariatene i denne modellen. Ettersom hvert individ altså kun bidrar til modellen gjennom sin risikoindeks er det mulig å bruke denne som et mål på hvor lenge pasienten vil leve. Dette gir altså ikke et estimat for predikert overlevelsestid, men ved å sammenlikne risikoindeks for flere individer er det mulig å si noe om en pasients overlevelse i forhold til andre individer.

En mulig strategi for å måle hvor godt en modell predikerer overlevelse er derfor å bruke et et feilmål som måler gjennomsnittlig feil i predikert risiko (GFPR). GFPR kan defineres som:

$$GFPR = \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{\beta}^T \mathbf{x}_i - \hat{\boldsymbol{\beta}}^T \mathbf{x}_i), \tag{3.12}$$

der n er antall individer, $\boldsymbol{\beta}$ er den sanne parametervektoren, $\hat{\boldsymbol{\beta}}$ er den estimerte parametervektoren og $\mathbf{x}_i$ er kovariatvektoren til individ i. Merk at dette feilmålet kun er mulig å bruke når den sanne parametervektoren er kjent.

Dersom man ser nærmere på uttrykket (3.12) ser man at det å beregne $GFPR$ tilsvarer å sammenlikne den sanne $\boldsymbol{\beta}$ med $\hat{\boldsymbol{\beta}}$ under en spesiell norm:

$$||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}||^2_{X^T X} = ||X(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}||^2 = ||X\boldsymbol{\beta} - X\hat{\boldsymbol{\beta}}||^2 = n * GFPR.$$

Dersom man studerer Cox-modellen ser man at et argument for å bruke $GFPR$ til å evaluere hvor godt $\boldsymbol{\beta}$ estimeres er at $\boldsymbol{\beta}$ kun inngår på formen $X\boldsymbol{\beta}$ i denne modellen. Man vil altså kunne

forvente at en god modell vil klare å finne et estimat $X\hat{\boldsymbol{\beta}}$ som ligger nær den sanne $X\boldsymbol{\beta}$. Dette vil likevel ikke nødvendigvis medføre at $\hat{\boldsymbol{\beta}}$ og $\boldsymbol{\beta}$ vil være like. Dette kan for eksempel komme av at hvis noen av kolonnene i X er kollineære, så ulike lineærkombinasjoner av kolonnene i X vil kunne gi den samme risikoen. Ved å bruke $GFPR$ istedenfor å beregne for eksempel $MSE(\hat{\boldsymbol{\beta}})$ (se appendiks A) tar man høyde for slike situasjoner, og får et bedre estimat for feil i predikert risiko. Ettersom det i denne studien er valgt å bruke stort sett bare simulerte data ble GFPR valgt som et av evalueringskriteriene for estimerte modeller. En annen grunn til dette er at det er et enkelt kriterium som er lett å beregne, selv for store datasett. Merk at selve verdien for GFPR kan være vanskelig å tolke, men den kan gi en mulighet for å skille mellom hvor godt en modell predikerer risiko i forhold til en annen modell.

Andre strategier for å benytte genekspresjonsdata i sammenheng med Cox-modellen for å predikere overlevelse kan finnes i blant annet Rosenwald et. al.[45], Beer et. al.[5], Lossos et. al.[36] og Bair og Tibshirani[4].

Kapittel 4

Regularisering

4.1 Innledning

For å kunne gjøre prediksjoner for nye individer, eller å kunne plukke ut signifikante gener med hensyn på overlevelse er det nødvendig å utføre en statistisk regresjon for å estimere en Cox-modell. I denne oppgaven vil en slik modell bli estimert på bakgrunn av mikroarraydata. Et mikroarraydatasett $X \in \mathbb{R}^{n \times p}$ består av målinger av ekspresjonen til p gener for n individer. Responsen består av overlevelsestider for disse n individene, med tilhørende sensurstatus for hver tid (se kapittel 3 for detaljer).

Standard algoritmer for tilpasning av Cox-modellen krever at antall individer skal være større enn antall kovariater, det vil si $n > p$. Helst bør n være flere ganger større enn p for at modellen skal kunne tilpasses direkte. Med data fra mikroarrayer har man derimot at $n << p$, det vil si at antall kovariater (gener) er mye større enn antall observasjoner (individer). Dette skaper en del problemer. Når $p > n$ vil rangen til datamatrisen X være mindre enn antall variable, $rank(X) < p$, ettersom man ikke kan ha p lineært uavhengige vektorer av lengde n. Dette resulterer i et såkalt *ill-posed problem*. Et problem er etter Hadamard [17] definert som *ill-posed* dersom løsningen til problemet ikke er unik, ikke eksisterer eller den ikke avhenger kontinuerlig av dataene, det vil si at små perturbasjoner i dataene kan føre til store perturbasjoner av løsningen. For å løse slike problemer må man gjøre det *well-posed* ved å legge restriksjoner på løsningene. Dette kalles *regularisering* av problemet, og det eksisterer flere teknikker for å gjøre det.

I dette kapittelet skal vi se nærmere på regularisering med særlig fokus på en teknikk som ofte benyttes og som anvendt på en lineær modell kalles *ridge regresjon*. Først i kapittelet kommer en kort innføring i ordinær lineær regresjon og singulærverdidekomposisjonen. Deretter følger en introduksjon til regularisering og straffet regresjon, før ridge regresjon vil bli gjennomgått. Vi skal se hvordan denne kan brukes på den klassiske lineære modellen og generelle lineære modeller som tilpasses ved hjelp av likelihoodbaserte metoder. Til slutt i kapittelet vil tilpasning av en spesiell klasse straffede modeller bli gjennomgått, der likelihooden er for en Cox-modell og straffen er en ridge-type straff.

4.2 Ordinær lineær regresjon

Vi starter med å se på situasjonen der vi ønsker å tilpasse en enkel lineær modell ved minste kvadraters metode. Som eksempel vil det brukes en designmatrise $X \in \mathbb{R}^{n\times p}$ som består av ekspresjonsdata fra mikroarrayer, men metodene er de samme for alle typer data. Som før angir n antall individer vi har målinger for, mens p er antall gener. Kovariatverdiene til person i representeres ved raden $\mathbf{x}_i$ i X, som inneholder genekspresjonsdata for p gener. Typisk for slike mikroarraydata er at $p >> n$, det vil si at for relativt få individer har man målt ekspresjonen til et mye større antall gener. For en gitt responsvektor $\mathbf{y} \in \mathbb{R}^n$ ønsker man å finne et uttrykk for sammenhengen mellom genekspresjonene og responsvektoren. En slik sammenheng kan skrives som

$$y_i = f(\mathbf{x}_i) + \epsilon, \quad E(\epsilon) = 0, \quad E(\epsilon_i, \epsilon_j) = \delta_{ij}\sigma^2, \tag{4.1}$$

der $f(\mathbf{x}_i)$ er en lineær funksjon av kovariatene, σ er variansen til støyleddet og vi antar at støyleddene er uavhengige slik at $\delta_{ij} = 1\ for\ i = j$ og 0 ellers. Et viktig teorem i statistikken som kalles *Gauss-Markov teoremet*[21] sier at minste kvadraters løsning for dette problemet, er det beste forventningsrette lineære estimatet blant alle forventningsrette lineære estimater. Dette tilsvarer å sette $\hat{f}(\mathbf{x}_i) = \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T\mathbf{x}_i$, der $\hat{\boldsymbol{\beta}} \in \mathbb{R}^p$. Estimatene $\beta_0, \hat{\boldsymbol{\beta}}$ finnes ved å minimere funksjonen

$$RSS(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2 = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2. \tag{4.2}$$

Det å gjøre en slik minste kvadraters tilpasning er ekvivalent med å gjøre det som kalles en lineær regresjon. Dersom X utvides til en $n\times(p+1)$-matrise med 1'ere i første kolonne vil man fra (4.2) kunne slå sammen konstantleddet β_0 med resten av uttrykket for koeffisientvektoren, slik at man får $RSS(\boldsymbol{\beta}) = \sum_{i=1}^{n}(y_i - \sum_{j=0}^{p}\beta_j x_{ij})^2$. Det vil videre i oppgaven være antatt at denne utvidelsen av X har blitt gjort i alle generelle lineære modeller der det inngår et konstantledd, dersom ikke annet er oppgitt. Legg merke til at løsningen fra nå av blir den utvidede vektoren $\hat{\boldsymbol{\beta}} \in \mathbb{R}^{(p+1)}$ der konstantleddet er tatt med.

Ved å utføre en lineær regresjon finner man den vektoren $\hat{\boldsymbol{\beta}} = (\beta_0, \beta_1, \ldots, \beta_p)^T$ som best tilpasser dataene med hensyn på de kvadrerte residualene, dvs. vi finner den vektoren $\hat{\mathbf{y}} = X\hat{\boldsymbol{\beta}}$ som ligger nærmest $\mathbf{y}$. Den mest naturlige måten å finne $\hat{\boldsymbol{\beta}}$ på i minste kvadraters metode er å benytte seg av *normallikningene*.

Teorem 4.2.1 (Normallikningene). *Dersom X har full kolonnerang er løsningen til* (4.2) *entydig definert. Løsningen er lettest å skrive på matriseform og kan da uttrykkes ved det som kalles for normallikningene:*

$$\hat{\boldsymbol{\beta}}_{OLS} = (X^TX)^{-1}X^T\mathbf{y}. \tag{4.3}$$

▶ **Bevis** . Vi skal gå gjennom et raskt bevis for hvordan man kan komme fram til denne løsningen. Ettersom X er utvidet til en $n \times (p+1)$-matrise med 1 i første kolonne, kan man skrive (4.2) på matriseform:

$$RSS(\boldsymbol{\beta}) = (\mathbf{y} - X\boldsymbol{\beta})^2 = (\mathbf{y} - X\boldsymbol{\beta})^T(\mathbf{y} - X\boldsymbol{\beta}). \tag{4.4}$$

For å finne minimum til (4.4) må uttrykket partiellderiveres med hensyn på komponentene i $\boldsymbol{\beta}$:

$$\frac{\partial RSS}{\partial \boldsymbol{\beta}} = -2X^T(\mathbf{y} - X\boldsymbol{\beta}).$$

Dette er en kvadratisk funksjon av de $p+1$ parameterne. Ettersom X er ikke-singulær (pga. full kolonnerang) blir X^TX positiv definitt og den førstederiverte kan dermed settes lik null:

$$X^T(\mathbf{y} - X\boldsymbol{\beta}) = 0,$$

og løsningen blir da

$$\hat{\boldsymbol{\beta}} = (X^TX)^{-1}X^T\mathbf{y},$$

som var det vi skulle vise.
□

Dersom man er i en situasjon der $n > p$ og (X^TX) fra teorem 4.2.1 er ikke-singulær, vil normallikningene kunne gi den entydige løsningen $\hat{\boldsymbol{\beta}}$ på problemet direkte. I noen situasjoner kan man derimot oppleve problemer med at X^TX blir singulær eller er dårlig kondisjonert. Hvis for eksempel $n < p$, vil $rang(X^TX) < n$ slik at X^TX blir singulær. Da har man et ill-posed problem, og normallikningene vil ikke være unikt definert og kan derfor ikke brukes til å finne en løsning direkte. En måte å håndtere slike problemer på er å bruke en variant av straffet regresjon som kalles ridge regresjon (se avsnitt 4.6. Før ridge regresjon introduseres skal vi se på singulærverdidekomposisjonen, et nyttig verktøy for å kunne analysere ill-posed problemer, og med det egenskaper ved ridge regresjon.

4.3 Singulærverdidekomposisjonen

Singulærverdidekomposisjonen er et av de beste og mest brukte verktøyene som finnes for å analysere problemer som ikke er av full rang, eller ill-posed problemer. Singulærverdidekomposisjonen vil først bli definert og så bli brukt til å gi en bedre forståelse av problemer som kan oppstå i situasjoner der vi ønsker å utføre en lineærregresjon når vi har en designmatrise $X \in \mathbb{R}^{n\times p}$ der $n < p$. Videre brukes dekomposisjonen til å analysere en mulig løsning for slike problemer, ved bruk av ridge regresjon.

Anta at vi har en matrise $X \in \mathbb{R}^{n\times p}$, og anta for enkelhets skyld først at $n \geq p$. Da kan X faktoriseres som $X = U\Sigma V^T$, der $U \in \mathbb{R}^{n\times p}$ og $V \in \mathbb{R}^{p\times p}$ begge har ortonormale kolonner (det vil si at $U^TU = V^TV = I_p$) og $\Sigma = diag(\sigma_1, \ldots, \sigma_p)$. Her er $\sigma_1, \ldots, \sigma_p$ singulærverdiene til X. Ettersom V er kvadratisk er den også rad-ortonormal (det vi si at $VV^T = I_p$). Denne dekomposisjonen kan illustreres ved:

$$X = \begin{pmatrix} x_{11} & \ldots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \ldots & x_{np} \end{pmatrix} = [\mathbf{u}_1\mathbf{u}_2\ldots\mathbf{u}_p] \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_p \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_p^T \end{pmatrix}.$$

Vektorene $\mathbf{u}_i \in \mathbb{R}^n$ kalles *venstresingulære vektorer* til X og vektorene $\mathbf{v}_i \in \mathbb{R}^p$ kalles *høyresingulære vektorer* til X. (Disse er også egenvektorene til kovariansmatrisen X^TX). Faktoriseringen $X = U\Sigma V^T$ kalles *singulærverdidekomposisjonen (SVD)* til X. Dersom $n < p$ kan man finne SVD til X ved å regne ut SVD til X^T og så transponere resultatet. Dette gir oss følgende teorem:

Teorem 4.3.1. *Enhver matrise $X \in \mathbb{R}^{n\times p}$ har en singulærverdidekomposisjon.*

Et bevis for dette teoremet kan finnes i [30]. Alle matriser X har altså en SVD, $X = U\Sigma V^T$. Denne kan være nyttig i mange situasjoner, blant annet når man ønsker å si noe om rangen til matrisen. Hvis man har $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ og $\sigma_{r+1} = \ldots \sigma_p = 0$, så er $rank(X) = r$. De r siste kolonnene i V utspenner da nullrommet til X, mens de r første kolonnene i U utspenner kolonnerommet til X. Matrisen Σ vil ha formen

$$\Sigma = \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix}, \tag{4.5}$$

der Σ_1 er diagonalmatrisen med $\sigma_1, \ldots, \sigma_r$ på diagonalen.

Et veldefinert gap mellom store og små singulærverdier er et tegn på at en eller flere av kolonnene i X er lineære kombinasjoner av de resterende kolonnene i X. Dersom de minste singulærverdiene til X er svært små kalles dette for *nær multikollinearitet*. Dette kommer klarere fram hvis man legger merke til at singulærverdidekomposisjonen til X kan skrives som en sum av ytreprodukter

$$X = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T. \tag{4.6}$$

Dersom man har at σ_r er nær 0, og definerer $\mathbf{a} = \mathbf{v}_r$ har man at $||X\mathbf{a}|| = \sigma_r \approx 0$, det vil si at $\sum_{i=1}^{p} a_i \mathbf{x}_i \approx 0$. Kolonnene i X er altså nesten lineært avhengige, og man sier da at kolonnene i X er nær multikollineære. Slik multikollinearitet er et tegn på at matrisen X ikke har full rang, eller at det er et ill-posed problem. Hvis alle singulærverdiene σ_i minker gradvis mot 0 er det også et ill-posed problem. En dimensjonsøkning av X vil generelt øke antall små singulærverdier, og dermed gjøre problemet ill-posed[18].

SVD kan også brukes til å analysere løsninger på lineære problemer. Anta at vi ønsker å løse systemet $\mathbf{y} = X\boldsymbol{\beta}$. Den opplagte løsningen $\boldsymbol{\beta} = X^{-1}\mathbf{y}$ er mulig å beregne hvis og bare hvis X er kvadratisk og har full rang. Anta nå at X ikke har full rang eller ikke er kvadratisk. Ved å bruke SVD av X kan man da definere den *pseudoinverse* til X som $X^\dagger = V\Sigma^\dagger U^T$, der $\Sigma^\dagger$ er definert som

$$\Sigma^\dagger = \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix},$$

ved å bruke definisjonen fra (4.5). En løsning på problemet $X\boldsymbol{\beta} = \mathbf{y}$ kan da skrives som en vektet sum av egenvektorene til X^TX:

$$\hat{\boldsymbol{\beta}} = X^\dagger \mathbf{y} = V\Sigma^\dagger U^T \mathbf{y} = \sum_{i=1}^{r} \frac{1}{\sigma_i} \langle \mathbf{u}_i, \mathbf{y} \rangle \mathbf{v}_i. \tag{4.7}$$

Legg merke til at en liten perturbasjon av $\mathbf{y}$ i en retning $\mathbf{u}_i$ med liten varians, det vil si liten σ_i, vil medføre en stor endring i $\hat{\boldsymbol{\beta}}$ på grunn av faktoren $\frac{1}{\sigma_i}$. Dette medfører at kovariater med liten varians vil bidra med mye varians i estimatoren $\hat{\boldsymbol{\beta}}$. Det er slike problemer man vil prøve å unngå ved å bruke regularisering.

4.4 Regularisering

En generell metodetilnærming til ill-posed problemer er det som kalles *regularisering*, eller *forventningsskjev regresjon* i regresjonssammenheng. Ideen bak dette er å erstatte det opprinnelige problemet med et penere, mer stabilt problem. Ved å legge en *straff* eller *betingelse* på fluktuasjonene i de estimerte parameterne til funksjonen som tilpasses, vil denne kunne bli glattere. Man velger den endelige løsningen som den funksjonen som gir best avveining mellom den sanne løsningen og numerisk stabilitet. Regularisering brukes innenfor en rekke områder, blant annet bildeanalyse og andre typer problemer som innebærer analyser av dårlig kondisjonerte data. Vi skal nå se hvordan regularisering kan innføres i en regresjonssammenheng, der funksjonen som estimeres bestemmes av en regresjonsvektor $\boldsymbol{\beta}$.

Så og si alle regulariseringsmetoder for regresjon er basert på krymping, det vil si at den opprinnelige uregulariserte løsningen $\hat{\boldsymbol{\beta}}$ fra en regresjon, erstattes med en regularisert løsning $\hat{\boldsymbol{\beta}}_{pen}$, som oppfyller $||\hat{\boldsymbol{\beta}}_{pen}|| < ||\hat{\boldsymbol{\beta}}||$, det vil si at vektoren $\hat{\boldsymbol{\beta}}_{pen}$ blir kortere enn den uregulariserte løsningsvektoren. En stor klasse av regulariseringsmetoder gir løsninger på formen:

$$\hat{\boldsymbol{\beta}} = \sum_{i=1}^{p} \omega_i \frac{\mathbf{u}_i^T \mathbf{y}}{\sigma_i} \mathbf{v}_i, \tag{4.8}$$

hvor ω_i er skalarer som bestemmer hvor mye løsningen skal krympes i forhold til minste kvadraters løsning langs hver av egenvektorene til $X^T X$, altså $\mathbf{v}_i$[34]. I minste kvadraters løsning er $\omega_i = 1$ for alle i. Hvor mye en regularisert løsning skal reguleres bestemmes av en såkalt *regulerings-* eller *straffeparameter*. For en regularisert løsning vil variansen til $\hat{\boldsymbol{\beta}}_{pen}$ minke jo større straff man legger på den. Samtidig blir den mer forventningsskjev ettersom straffen øker. Ideen er at man kan finne en optimal størrelse på straffeparameteren, som gjør at man finner et estimat $\hat{\boldsymbol{\beta}}_{pen}$ som ligger nærmere den virkelige løsningen $\boldsymbol{\beta}$ enn den uregulariserte løsningen, dvs. $||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}_{pen}|| < ||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}||$. Man ønsker altså å finne en bedre estimator ved å gjøre en avveining mellom økt bias og mindre varians.

Det finnes to hovedteknikker innenfor regularisering. Den ene er projeksjonsteknikker. Slike metoder prøver å gjøre en dimensjonsreduksjon av kovariatdataene ved å projisere de opprinnelige høydimensjonale kovariatdataene X ned i et lavere dimensjonalt underrom, og man lager en matrise Z som representerer projeksjonen av X ned på dette underrommet. Så tilpasses en modell med Z som kovariatmatrise, mens responsen beholdes uendret. Hvordan man finner dette underrommet varierer mellom de ulike metodene. *Prinsipal komponent regresjon (PCR)* finner de ortogonale retningene i data som har mest varians, *prinsipalkomponentene*, og projiserer data ned på disse retningene. For PCR vil vektene i (4.8) ta verdiene $\omega_i = 1$ for $i \leq m$ og $\omega_i = 0$ for $i > m$, der m er antall prinsipalkomponenter man inkluderer i modellen. Ideen er å ta med så mange prinsipalkomponenter at man kan forklare så mye av variansen i responsen som mulig, uten å ta med for mye unødvendig støy. Antagelsen som gjøres er at kovariater med liten varians ikke vil bidra vesentlig til responsen, og om de taes med vil de bidra mer med støy enn med nyttig infomasjon. Det er trivielt å se at denne metoden vil gi et estimat $\hat{\boldsymbol{\beta}}_{PCR}$ som har mindre norm enn minste kvadraters metode, ettersom PCR i uttrykket (4.8) enten tar med like mange eller færre ledd enn minste kvadraters løsning, og dermed vil $||\hat{\boldsymbol{\beta}}_{PCR}|| \leq ||\hat{\boldsymbol{\beta}}||$.

Partial least squares (PLS) er en dimensjonsreduksjonsmetode på linje med PCR, men den har

en betydelig mer komplisert krympingsstruktur, se for eksempel Lingjærde og Christoffersen[34]. De viser blant annet at for PLS-estimatet får vektene fra (4.8) en noe mer komplisert struktur. For PLS med m faktorer får vektene formen

$$\omega_i^{(m)} = 1 - \prod_{j=1}^{m}(1 - \frac{\lambda_i}{\theta_j^{(m)}}) \text{ for } i = 1, \ldots, p,$$

der $\lambda_1 \geq \ldots \geq \lambda_p$ er egenverdiene til $X^T X$ og $\theta_1^{(m)} \geq \ldots \theta_p^{(m)}$ er egenverdiene til matrisen $V_m^T X^T X V_m$. Matrisen V_m er her en hvilken som helst $p \times m$-matrise med kolonner som danner en ortonormal basis for Krylov-rommet $K_m = span(X^T\mathbf{y}, (X^T X)X^T\mathbf{y}, \ldots, (X^T X)^{m-1}X^T\mathbf{y})$. Den kompliserte krympingsstrukturen til PLS gjør at noen av komponentene i $\hat{\boldsymbol{\beta}}_{PLS}$ kan øke i forhold til $\hat{\boldsymbol{\beta}}$, det vil si at noen av vektene ω_i kan bli større enn 1, mens andre blir mindre. Det er likevel mulig å vise at estimatet $\hat{\boldsymbol{\beta}}_{PLS}$ tilfredsstiller kravet for en regularisert løsning, det vil si at $||\hat{\boldsymbol{\beta}}_{PLS}|| \leq ||\hat{\boldsymbol{\beta}}||$[10, 15]. PLS ble i utgangspunktet utviklet for lineære modeller, og brukes mye innen kjemometri. Den kan generaliseres til ulike andre situasjoner, for eksempel til Cox regresjon og generaliserte lineære modeller, se Park et al.[40] og Nguyen og Rocke[38] for eksempler på dette. Projeksjonsmetoder velger altså ut et subsett av prediktorer, og dropper resten. De er med det såkalte diskrete prosesser, i og med at bidraget fra en variabel enten tas med eller ikke. Med dette prøver de å oppnå en bedre modell enn den basert på det fulle datasettet. Ulempen er at disse metodene i noen tilfeller kan resultere i estimater med høy varians, og dermed gi større prediksjonsfeil enn en uregularisert regresjon.

Den andre hovedteknikken innenfor regularisering er det som kalles *kontinuerlige krympingsmetoder*. Disse metodene fungerer ved å legge en straff på regresjonskoeffisienten $\boldsymbol{\beta}$ slik at komponentene i $\boldsymbol{\beta}$ krympes mot 0 og mot hverandre. Fordelen med dette er at man ikke velger bort noen variable, slik at den totale variansen reduseres kontinuerlig. Ved å bruke slike krympingsmetoder kan man unngå noen av problemene som kan oppstå ved bruk av PLS eller PCR, og dermed få bedre resultater. Ridge regresjon er et eksempel på en slik kontinuerlig krympingsmetode. For ridge regresjon vil vektene fra (4.8) ha formen $\omega_i = 1/(1 + (\lambda/\sigma_p^2))$, $i = 1, \ldots, p$, der λ er parameteren som regulerer straffen. Vi vil komme tilbake til dette resultatet senere under presentasjonen av ridge regresjon i avsnitt 4.6. Ridge regresjon er en form for straffet regresjon, og vi skal nå se nærmere på denne typen regresjon.

4.5 Straffet regresjon

Det å gjøre en straffet regresjon vil si at man legger en straff eller betingelse på regresjonskoeffisientene, β_i, slik at de krympes mot 0 etter en lineær funksjon av parametervektoren $\boldsymbol{\beta}$. For likelihoodbaserte modeller vil dette kunne gjøres ved hjelp av å estimere koeffisientene ved en straffet likelihood. Etter O'Sullivan[39] får man en generell straffet log-likelihood ved å ta en generell straffefunksjon, $g_\lambda(\cdot)$, og trekke denne fra log-likelihooden. Dette kan skrives

$$\min_{\beta_0, \boldsymbol{\beta}} = \sum_{i=1}^{n} l(y_i, x_i|\beta_0, \boldsymbol{\beta})) - \sum_{i=1}^{p} g_\lambda(\boldsymbol{\beta}). \tag{4.9}$$

Her er $l(\cdot)$ en kjent log-likelihoodfunksjon. Straffefunksjonen $g_\lambda(\cdot)$ kan ta ulike former, og det finnes en rekke slike straffefunksjoner med ulike egenskaper [21, 26, 56, 51]. Parameteren λ

blir i denne sammenhengen kalt en *tuningparameter*, og den regulerer hvor mye koeffisientene blir straffet. Jo større λ er, jo mer krympes koeffisientene mot 0. Dersom man antar at λ er en konstant som holdes fast kan en slik straffet likelihood optimeres ved vanlige ML-metoder, slik at man finner den $\hat{\boldsymbol{\beta}}_{pen}$ som optimerer den straffede likelihooden. Det er også mulig å finne den λ som maksimerer den straffede likelihooden ved å bruke ulike modellseleksjonsmetoder, som for eksempel kryssvalidering. Slike metoder vil bli presentert i kapittel 6. For enkelhets skyld kan vi anta i dette kapittelet at λ er en konstant som holdes fast.

Videre i kapittelet vil det bli presentert en form for straffet regresjon som kalles *kvadratisk regularisering* eller *ridge regresjon*, som kan brukes til å løse ill-posed problemer. Vi starter med å se på ordinær ridge regresjon for en lineær modell. Deretter vil disse teknikkene bli overført til en generell lineær modell, der modelltilpasningen baseres på likelihoodestimering. Som et spesialtilfelle av dette skal vi se på en straffet Cox-modell, og se at man ved hjelp av denne metoden kan finne estimater for $\boldsymbol{\beta}$ i Cox-modellen som både er mer stabile og som kan gi bedre prediksjonsresultater enn estimatene beregnet med den ustraffede log-likelihooden(3.10).

4.6 Ridge Regresjon

Ridge regresjon ble introdusert av Hoerl og Kennard i 1970[23] som en mulig håndtering av problemer med multikollinearitet som kan oppstå under lineær regresjon. Målet med metoden er å forskyve løsningsvektoren $\hat{\boldsymbol{\beta}}$ bort fra retninger der prediksjonsvektorene, $\mathbf{x}_i$, har liten spredning og ved dette oppnå mer stabile regresjonsestimater i situasjoner med høy kollinearitet. Hoerl og Kennard viste at ved å forskyve løsningsvektoren $\hat{\boldsymbol{\beta}}$ litt er det mulig å forbedre prediksjonen, ettersom denne forskyvningen vil kunne medføre en reduksjon av variansen. Ridge regresjon krymper regresjonskoeffisientene ved å legge en straff på størrelsen deres, og parameteren λ kontrollerer hvor mye det skal straffes. Vi skal starte med å se på et resultat fra Hoerl og Kennard som viser en av egenskapene ved ridge regresjon som gjør at den straffede løsningen i noen tilfeller kan være bedre enn den fra ordinær lineær regresjon. Deretter går vi inn i detaljene for hvordan ridge regresjon utføres.

Hoerl og Kennard observerte at den totale variansen til en straffet regresjonsvektor $\hat{\boldsymbol{\beta}}_\lambda$ minket med økende straff, mens den kvadrerte biasen til regresjonsvektoren økte. Som beskrevet i appendiks A kan feilen mellom den estimerte og den sanne $\boldsymbol{\beta}$ kan dekomponeres til en sum av varians og kvadrert bias, $MSE(\hat{\boldsymbol{\beta}}(\lambda)) = var(\hat{\boldsymbol{\beta}}(\lambda)) + bias(\hat{\boldsymbol{\beta}}(\lambda))^2$.

Ved å definere den kvadrerte avstanden mellom den sanne $\boldsymbol{\beta}$ og det straffede estimatet $\hat{\boldsymbol{\beta}}(\lambda)$ for en verdi λ som

$$L(\lambda)^2 = ||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}(\lambda)||, \tag{4.10}$$

kan man finne forventet avstand mellom disse som

$$E[L(\lambda)^2] = E(||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}(\lambda)||) = MSE(\hat{\boldsymbol{\beta}}(\lambda)).$$

Det å sette $\lambda = 0$, det vil si ridge regresjon uten straff, tilsvarer å beregne OLS-estimatoren, $\hat{\boldsymbol{\beta}}_{OLS}$ fra (4.3). Hoerl og Kennard viste at det er mulig å finne en $\lambda > 0$ som gjør at avstanden (4.10) blir mindre enn for OLS-estimatoren. De kaller dette resultatet for *ridge eksistens teorem*. Det sier at det alltid eksisterer en $\lambda > 0$ som gjør at

$$E[L(\lambda)^2] < E[L(0)^2] \iff MSE(\hat{\boldsymbol{\beta}}(\lambda)) < MSE(\hat{\boldsymbol{\beta}}_{OLS}).$$

Det vil si at det alltid er mulig å finne en løsningsvektor som ligger nærmere den sanne vektoren enn minste kvadraters estimatet (OLS-estimatet). Se [23] for nærmere detaljer rundt dette resultatet.

4.6.1 Ridge regresjon for ordinære lineære modeller

Vi har sett tidligere at i tilfeller der matrisen X^TX er singulær eller nesten singulær er det ikke mulig å finne en løsning på det lineære problemet $\mathbf{y} = X\boldsymbol{\beta}$ ved å bruke normallikningene. Det viser seg derimot at ved å legge et multiplum av identitetsmatrisen I til X^TX i normallikningene kan løsningen stabiliseres. Dette kalles *ridge regresjon*, eller *ridge straff*, ettersom effekten av å gjøre dette er at man straffer løsningsvektoren ved å gjøre den mindre. Graden av stabilisering reguleres av den såkalte *ridgeparameteren* λ. Ridge estimatet $\hat{\boldsymbol{\beta}}(\lambda)$ er vektoren som minimerer uttrykket

$$RSS(\lambda) = \sum_{i=1}^{n}(y_i - \boldsymbol{\beta}^T\mathbf{x}_i)^2 + \lambda\sum_{j=1}^{p}\beta_j^2. \tag{4.11}$$

Legg merke til at dette er den samme likningen som løsningen for den ordinære lineære regresjonen (4.2), bortsett fra at man har lagt til et ekstra straffeledd $\lambda\sum_{j=1}^{p}\beta_j^2$. Skrives dette på matriseform er det lettere å se hvilken form løsningen på det straffede problemet får:

$$RSS(\lambda) = (\mathbf{y} - X\boldsymbol{\beta})^T(\mathbf{y} - X\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}. \tag{4.12}$$

Ved å minimere denne med hensyn på $\boldsymbol{\beta}$ på samme måte som det ble gjort for (4.4) får man at løsningen for ridge regresjon blir

$$\hat{\boldsymbol{\beta}}(\lambda) = (X^TX + \lambda I)^{-1}X^T\mathbf{y}. \tag{4.13}$$

Dette estimatet plugger man så inn i den ordinære lineære modellen fra (4.1) og den estimerte modellen blir:

$$\hat{y_i} = \hat{f}_\lambda(\mathbf{x}_i) = \hat{\boldsymbol{\beta}}(\lambda)^T\mathbf{x}_i.$$

Legg merke til at den estimerte modellen er avhengig av λ i og med at den er basert på det straffede estimatet $\hat{\boldsymbol{\beta}}(\lambda)$. Modellen kan tolkes og brukes til prediksjon som en ordinær lineær modell.

Ved å bruke ridge regresjon vil det alltid være mulig å finne en løsning på ill-posed problemer. Dette kan man se ut fra at den inverse $(X^TX + \lambda I)^{-1}$ fra (4.13) aldri vil være singulær for $\lambda > 0$. Dermed unngår man problemene som oppstår med at X^TX er singulær. For å kunne forstå hvorfor det blir slik kan det være til hjelp å uttrykke ridge-løsningen (4.13) ved hjelp av singulærverdidekomposisjonen $X = U\Sigma V^T$, som ble definert i avsnitt 4.3. Dette gir følgende uttrykk:

$$\begin{aligned}\hat{\boldsymbol{\beta}}(\lambda) &= (V\Sigma U^TU\Sigma V^T + \lambda I)^{-1}V\Sigma U^T\mathbf{y} \\ &= V(\Sigma^2 + \lambda I)^{-1}V^TV\Sigma U^T\mathbf{y} \\ &= V(\Sigma^2 + \lambda I)^{-1}\Sigma U^T\mathbf{y} \end{aligned} \tag{4.14}$$

Vektoren med de estimerte responsverdiene blir da

$$\hat{\mathbf{y}} = X\hat{\boldsymbol{\beta}}(\lambda) = U\Sigma^*U^T\mathbf{y}$$

der

$$\Sigma^* = \Sigma(\Sigma^2 + \lambda I)^{-1}\Sigma = \begin{pmatrix} \frac{1}{1+(\lambda/\sigma_1^2)} & & \\ & \ddots & \\ & & \frac{1}{1+(\lambda/\sigma_p^2)} \end{pmatrix}$$

Ut fra disse likningene kan man se at estimatoren $\hat{\boldsymbol{\beta}}(\lambda)$ gjør at $\mathbf{y}$ vektes ned i alle retninger, og at størrelsen på nedvektingen bestemmes av størrelsen på singulærverdiene, σ_i, $i = 1, \ldots, p$. Diagonalelementene i Σ^* kalles *krympingsfaktorer* og gjør at $\mathbf{y}$ krympes mest i retningene med små singulærverdier. Metoden krymper altså automatisk variable med minst varians mest, slik at mest mulig støy fjernes fra dataene. Variable med høy varians krympes mindre slik at mest mulig av bidraget fra disse komponentene beholdes. Ridge regresjon involverer altså å skalere $\boldsymbol{\beta}^T\mathbf{x}$ etter hvor stor varians det er i observasjonene, for slik å produsere forventningsskjeve estimater, det vil si at $E[\hat{\boldsymbol{\beta}}(\lambda)] \neq \boldsymbol{\beta}$. Hvor mye bias som innføres reguleres av straffeparameteren λ. Hvis $\lambda > 0$ innføres det et økt bias mot større verdier av $var(\boldsymbol{\beta}^T\mathbf{x})$, og økt krymping av lengden til løsningsvektoren. For små verdier av λ vil den førstnevnte effekten være størst. Metoden krymper alle koeffisientene kontinuerlig mot 0 for økende verdier av λ, men den setter ingen av koeffisientene eksakt til 0. Ved å bruke ridge regresjon får man altså krympede komponenter av $\boldsymbol{\beta}$, men den utfører ingen automatisk variabelseleksjon i og med at ingen av komponentene settes eksakt lik 0. Dersom man ønsker å teste hvilke av koeffisientene som er signifikante må dette gjøres ved hypotesetesting.

Generalisert ridge regresjon

Det er mulig å utføre en generalisert variant av ridge regresjon, der man har muligheten til å straffe de ulike komponentene av $\boldsymbol{\beta}$ individuelt. Dette kan være ønskelig dersom man for eksempel har forhåndsinformasjon om dataene som sier at noen av kovariatene er viktigere enn andre. Har man for eksempel et mikroarraydatasett der man har stor tiltro til at noen utvalgte gener kommer til å være viktige for responsen kan man velge å gi disse genene lavere straff enn resten av genene i datasettet, slik at man unngår å straffe disse viktige genene for mye. I en generalisert ridge regresjon får straffeleddet fra 4.11 formen $g_\lambda(\boldsymbol{\beta}) = \sum_{j=1}^{p} \lambda_j \beta_j^2$, der λ_j er den individuelle straffen til kovariat j. Dette straffeleddet kan også skrives på matriseform $\boldsymbol{\beta}^T Q \boldsymbol{\beta}$, der Q er en diagonalmatrise med verdiene $(\lambda_1, \ldots, \lambda_p), \lambda_i > 0,\ i = 1, \ldots, p$, på diagonalen. Løsningen $\hat{\boldsymbol{\beta}}(Q)$ blir:

$$\hat{\boldsymbol{\beta}}(Q) = (X^T X + Q)X^T\mathbf{y}$$

4.6.2 Bayesiansk utledning av ridge regresjon

Ridge regresjon kan også motiveres med bakgrunn i Bayesiansk teori. Dette ble også poengtert av Hoerl og Kennard [23], og vi skal gå kort gjennom dette resonnementet her og se at ridge regresjon har en naturlig kobling til bayesiansk statistikk. Vi tar utgangspunkt i den standard lineære regresjonsmodellen:

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

der $\mathbf{y} \in \mathbb{R}^n$ er vektoren med responser, $X \in \mathbb{R}^{n \times p}$ er matrisen med kovariatverdiene og $\boldsymbol{\beta} \in \mathbb{R}^p$ er vektoren med regresjonsparametere som man ønsker å estimere. Dersom man antar at støyleddene ϵ_i er uavhengig trukket fra normalfordelingen $N(0, 1)$ og at observasjonene $\mathbf{x}_i$

også er uavhengige, vil dette medføre at responsene y_i vil være uavhengig normalfordelte med forventning $\boldsymbol{\beta}^T\mathbf{x}_i$ og varians 1, dvs. $y_i \sim N(\boldsymbol{\beta}^T\mathbf{x}_i, 1)$.

Anta så at $\boldsymbol{\beta}$ har en *priorfordeling* $\boldsymbol{\beta} \sim N_p(0, Z)$, der Z er kovariansmatrisen. I følge bayesiansk teori (se appendiks A) kan man da finne *a posteriorifordelingen* $f(\boldsymbol{\beta}|\mathbf{y})$ ved å bruke Bayes formel. Dette uttrykket kan skrives som

$$f(\boldsymbol{\beta}|\mathbf{y}) = \frac{f(\boldsymbol{\beta})\ f(\mathbf{y}|\boldsymbol{\beta})}{f(\mathbf{y})}. \tag{4.15}$$

Ettersom man i praksis beregner fordelingen til $\boldsymbol{\beta}$ på bakgrunn av en fast observert respons $\mathbf{y}$ vil fordelingen $f(\mathbf{y})$ bare bli en skaleringsfaktor bestemt av den observerte $\mathbf{y}$. Dette medfører at

$$f(\boldsymbol{\beta}|\mathbf{y}) \propto f(\boldsymbol{\beta})\ f(\mathbf{y}|\boldsymbol{\beta}).$$

Tettheten til en multivariat normalfordeling $N_d(\boldsymbol{\mu}, \Sigma)$, der kovariansmatrisen Σ med elementer σ_{ij} er ikke-singulær, er som kjent uttrykt ved:

$$f(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} e^{[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})]},$$

der $|\Sigma|$ er determinanten til kovariansmatrisen. Dersom alle observasjonene er uavhengige, det vil si at $\sigma_{ij} = 0$ for $i \neq j$, blir Σ en diagonalmatrise og tettheten $f(\mathbf{x})$ reduseres til produktet av den univariate normaltettheten til komponentene i $\mathbf{x}$. Med bakgrunn i dette kan man skrive (4.15) som

$$\begin{aligned} f(\boldsymbol{\beta}|\mathbf{y}) &\propto \frac{1}{\sqrt{(2\pi)^d|Z|}} e^{[-\frac{1}{2}\boldsymbol{\beta}^T Z^{-1}\boldsymbol{\beta}]} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} e^{[-\frac{1}{2}(y_i-\boldsymbol{\beta}^T\mathbf{x}_i)^2]} && (4.16)\\ &\propto e^{[-\frac{1}{2}\boldsymbol{\beta}^T Z^{-1}\boldsymbol{\beta}]} \prod_{i=1}^{n} e^{[-\frac{1}{2}(y_i-\boldsymbol{\beta}^T\mathbf{x}_i)^2]} && (4.17)\\ &= e^{[-\frac{1}{2}(\boldsymbol{\beta}^T\Sigma^{-1}\boldsymbol{\beta}+\sum_{i=1}^{n}(y_i-\boldsymbol{\beta}^T\mathbf{x}_i)^2]}. && (4.18) \end{aligned}$$

Dersom man antar at kovariansmatrisen i priorfordelingen til $\boldsymbol{\beta}$ er diagonalmatrisen $\Sigma = \lambda^{-1}I$, det vil si at alle komponentene i $\boldsymbol{\beta}$ er uavhengige og har lik varians λ^{-1}, vil uttrykket over kunne skrives som

$$e^{[-\frac{1}{2}(\lambda||\boldsymbol{\beta}||^2+\sum_{i=1}^{n}(y_i-\boldsymbol{\beta}^T\mathbf{x}_i)^2]}.$$

Det å finne en $\boldsymbol{\beta}$ som maksimerer dette uttrykket er det samme som å finne en $\boldsymbol{\beta}$ som minimerer

$$\sum_{i=1}^{n}(y_i - \boldsymbol{\beta}^T\mathbf{x}_i)^2 + \lambda||\boldsymbol{\beta}||^2.$$

Dette er det samme uttrykket som vi hadde for ridge regresjon (4.13). Det å anta priorfordelingen $\boldsymbol{\beta} \sim N(\mathbf{0}, \lambda^{-1}I)$ for regresjonsparameterne blir altså ekvivalent med å gjøre en ridge regresjon med straffeparameter λ. Man kan vise [27, 32] at *a posteriori* fordelingen til $\boldsymbol{\beta}$ blir

$$f(\boldsymbol{\beta}|\mathbf{y}) \sim N((X^TX + \lambda I)^{-1}X^TY, (X^TX + \lambda I)^{-1}).$$

A posteriori forventningen til $\boldsymbol{\beta}$ blir altså lik uttrykket for ridge estimatet $\hat{\boldsymbol{\beta}}(\lambda)$ fra (4.13). Ved å gjøre en rimelig antagelse om at $\boldsymbol{\epsilon}$ er normalfordelt med forventning $\mathbf{0}$ og varians I så

kan man se på ridge estimatoren som *a posteriori* forventningen basert på en normalfordelt priorfordeling for $\boldsymbol{\beta}$ med forventning $\mathbf{0}$ og varians $\lambda^{-1}I$.

Ut fra dette kan man se at for at ridge regresjon skal ha fornuftige resultater må man kunne forvente at alle komponentene i $\boldsymbol{\beta}$ er identisk uavhengig normalfordelte med forventning 0. Dersom man for eksempel vet at noen av komponentene i $\boldsymbol{\beta}$ aldri kan bli negative så vil ikke lenger ridge estimatet være gyldig. Man kan derimot utvide ridge estimatet til slike mer generelle situasjoner, der forventningen til $\boldsymbol{\beta}$ kan være ulik $\mathbf{0}$[27]. Detaljene runde dette vil ikke bli gjennomgått her. Dersom man ønsker å straffe de ulike komponentene i $\boldsymbol{\beta}$ forskjellig ved generalisert ridge regresjon, presentert i avsnitt 4.6, der straffeleddet var på formen $\boldsymbol{\beta}^T Q\boldsymbol{\beta}$, blir dette ekvivalent med å anta at priorfordelingen har varians $Z = diag(\lambda_1, \ldots, \lambda_p)I$.

Vi har sett at ridge regresjon kan være nyttig for ordinær lineær regresjon når vi har ill-posed problemer. Vi skal nå se videre hvordan disse ideene kan overføres til mer generelle lineære modeller, og gi mer robuste og stabile estimatorer også i disse tilfellene.

4.6.3 Ridge regresjon for modeller med lineære prediktorer

I en rekke modeller involveres kovariatene gjennom en lineær prediktor. Eksempler på slike modeller er logistisk regresjon, Cox proporsjonale modell, diskriminantanalyse, generaliserte lineære modeller og nevrale nett. Felles for alle disse er at de involverer kovariatene $\mathbf{x}$ gjennom en eller flere lineære funksjoner, det vil si på formen $\boldsymbol{\beta}^T\mathbf{x}$. Slike modeller tilpasses ved å minimere en loss-funksjon L som kan være kvadrert feil, negativ log-likelihood, negativ partiell log-likelihood og liknende. Vanligvis krever en slik tilpasning at man har flere observasjoner enn variable. Dersom antall observasjoner blir mindre enn antall variable vil det oppstå problemer dersom man prøver å tilpasse ved bruk at standard metoder.En måte å kompensere for dette problemet er å innføre en kvadratisk regularisering på loss-funksjonen. Dette gjøres ved å trekke et kvadratisk straffeledd fra loss-funksjonen, på tilsvarende måte som det ble gjort i ordinær ridge regresjon. Det man kan oppnå med dette er å få en glattere loss-funksjon, og dermed mer stabile estimater for $\boldsymbol{\beta}$. Dersom man velger å straffe alle komponentene i $\boldsymbol{\beta}$ likt kan straffeleddet skrives som $\lambda||\boldsymbol{\beta}||_2^2$, og hele det straffede uttrykket får formen:

$$\min_{\boldsymbol{\beta}} = \sum_{i=1}^{n} L(y_i, \mathbf{x}_i^T\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}. \tag{4.19}$$

Dette uttrykket blir en kvadratisk funksjon av koeffisientene. For en fornuftig verdi av λ vil det være mulig å finne et straffet estimat $\hat{\boldsymbol{\beta}}(\lambda)$, som man til slutt kan bruke til å tilpasse den endelige modellen.

I denne oppgaven vil vi konsentrere oss om en spesiell likelihood med kvadratisk straff, nemlig den straffede partielle likelihooden til Cox-modellen som ble introdusert i kapittel 3.

4.7 Den straffede Cox modellen

Dersom antall observasjoner blir større enn antall variable vil den partielle likelihooden til Cox-modellen bli så ustabil at det ikke er mulig å finne et maksimum likelihoodestimat $\hat{\boldsymbol{\beta}}$

for denne modellen. Ved å innføre en kvadratisk straff på den partielle log-likelihooden til Cox-modellen(3.10) kan man unngå slike problemer. Den straffede partielle log-likelihooden blir på formen

$$l_{pen}(\boldsymbol{\beta}, \lambda) = l(\boldsymbol{\beta}) - \sum_{j=1}^{p} \lambda_j \beta_j^2 = 0, \tag{4.20}$$

der $l(\boldsymbol{\beta})$ er den partielle log-likelihooden og λ er straffeparameteren. En slik straffet log-likelihood kan tilpasses på samme måte som vi har sett tidligere ved hjelp av maksimum likelihoodmetoder (se appendiks A). Dette gjøres ved å sette de partiellderiverte av den straffede log-likelihooden med hensyn på komponentene i $\boldsymbol{\beta}$ lik 0, det vil si at man løser de p likningene

$$\frac{\partial l_{pen}}{\partial \beta_q} = \sum_{i=1}^{n} \mathbf{x}_{iq} - \sum_{i=1}^{n} \frac{\sum_{j \in R_{t_i}} \mathbf{x}_{jq} exp(\boldsymbol{\beta}^T \mathbf{x}_j)}{\sum_{j \in R_{t_i}} exp(\boldsymbol{\beta}^T \mathbf{x}_j)} - \sum_{j=1}^{p} \lambda_j \beta_j^2,$$

for $q = 1, \ldots, p$. Det er ikke mulig å finne eksplisitte uttrykk for β_q ut fra disse likningene, så maksimum likelihoodestimatet finnes ved å bruke en iterativ optimeringsalgoritme. Hvordan dette gjøres kommer vi tilbake til i kapittel 5. For å velge en verdi for straffeparameteren λ som gjør at vi får gode estimater for $\boldsymbol{\beta}$ kan man bruke ulike metoder for modellseleksjon. Hva som menes med gode estimater og hvordan man kan bestemme dette, vil vi se nærmere på i kapittel 6.

4.8 Andre tilnærminger

Mange av analysene som til nå er gjort av genekspresjonsdata er basert på klassifiseringsteknikker, det vil si at man deler pasienter eller gener inn i ulike grupper etter liknende genekspresjonsmønstre. I Dudoit et al. [13] finnes en oppsummering over mye av det som er gjort innen klassifisering. Ulike tilnærminger har vært foreslått for å knytte sensurerte overlevelsesdata opp mot ekspresjonsdata, både med den hensikt å kunne plukke ut gener med ekspresjonsmønstre som er assosiert med overlevelsestider, og for å kunne bygge statistiske modeller som kan brukes til prediksjon av overlevelse. Mange av disse arbeidene har involvert bruk av regulariseringsmetoder. Noen arbeider vil omtales kort nedenfor.

Hastie et al.[20] presenterte i 2000 en metode kalt *gene shaving* for å kunne plukke ut gener som er korrelert med ulike mulige responstyper, blant annet overlevelse. Metoden baseres på klustring av data som blant annet gjør det mulig å velge ut grupper av korrelerte gener. Resultatene av denne metoden er derimot sensitive for hvilken klustringsmetode som benyttes.

Nguyen og Rocke[38] presenterte i 2002 et forslag der de bruker dimensjonsreduksjonsmetoden *partial least squares* (PLS) direkte på overlevelsesdataene (se for eksempel [21] og [34] for detaljer rundt PLS). De bruker så PLS-komponentene som kovariater i Cox proporsjonale hasardmodell for å predikere overlevelse. Denne tilnærmingen benytter PLS direkte på overlevelsesdataene uten å ta hensyn til forskjellen mellom tid til hendelse og tid til sensurering (se kapittel 3).

Ulike varianter av utvelgelse av gener på bakgrunn av univariat Cox-regresjon har også blitt gjort, se blant annet Beer et al.[5] og Lossos et al. [36]. Prediksjon av overlevelse blir her

basert på et lite subsett av genene som gjennom univariat regresjon viser sterkest relasjon til overlevelsestidene. Metodene har gitt forholdsvis gode resultater, men kan muligens overse relevant informasjon ved at man forkaster mange gener under utvelgelsen.

Park et al.[40] foreslo i 2002 en måte å unngå hele problemet med sensurerte data ved å omformulere Cox-modellen til et Poisson-regresjonsproblem. Videre ble PLS brukt for å danne lineære kombinasjoner av genekspresjonsvektorene, som igjen ble brukt som kovariater i regresjonen. Ulempen med en slik omformulering er at man introduserer en rekke nye forstyrrende parametere. Metoden ble i dette arbeidet heller ikke testet ut for hvor godt den presterer i prediksjon for nye individer.

Li og Luan[31] var de første til å diskutere L_2-straffet estimering av Cox-modellen basert på teorier for reproduserende kjerner. Prosedyren reduseres til en standard Cox-modell med L_2-straff når indreproduktkjernen benyttes. De viste blant annet at en slik tilnærming kan brukes til å lage en modell for prediksjon av levetider, men de presenterer derimot ingen praktisk tilnærming for å kunne velge mellom ulike kjernefunksjoner eller korresponderende tuningparametere.

Kapittel 5

Optimering

[1] Vi ønsker å kunne optimalisere den straffede log-likelihoodfunksjonen til Cox-modellen (4.20) for å kunne finne et estimat for regresjonsvektoren $\boldsymbol{\beta}$. Dette kan gjøres ved bruk av standard kalkulus ved å sette de partiellderiverte lik 0, det vil si $\partial l/\partial \boldsymbol{\beta} = 0$ (se appendiks A). Fra uttrykket for de partiellderiverte til Cox-modellen er det ikke mulig å finne eksplisitte uttrykk for $\hat{\boldsymbol{\beta}}$, men det er derimot mulig å finne maksimum av funksjonen ved å benytte en iterativ optimeringsalgoritme. En slik algoritme finner optimum til en funksjon i k iterasjoner ved å bruke informasjon om funksjonen, og eventuelt dens deriverte, i hver iterasjon.

Anta i dette kapittelet at f er generell multivariat funksjon av p variable som man ønsker å minimere. Dersom man ønsker å maksimere f blir dette ekvivalent med å minimere den negative funksjonen, $-f$. Kort fortalt vil iterative metoder for optimering starte med et initialt estimat $\mathbf{x}_0$ for løsningen $\mathbf{x}^*$, og generere en følge $\{\mathbf{x}_k\}$ av estimater for $\mathbf{x}^*$, der $\mathbf{x}_k$ er en bedre approksimasjon til løsningen $\mathbf{x}^*$ enn den forrige approksimasjonen $\mathbf{x}_{k-1}$. Noen iterative metoder benytter seg bare av informasjon om funksjonen f i hver iterasjon, mens andre også bruker informasjon om de deriverte til f.

Vi skal se kort på et par ulike typer iterative optimeringsalgoritmer, og et par eksempler på disse, før vi skal se nærmere på detaljene rundt metoden som ble brukt i denne oppgaven.

5.1 Direkte søkemetoder

Direkte søkemetoder bruker kun informasjon om f i hver iterasjon. I hver iterasjon sammenliknes verdien av f i en mengde ulike punkter. Det beste av disse punktene velges som den neste approksimasjonen, inntil en endelig approksimasjon til løsningen $\mathbf{x}^*$ er funnet. Et eksempel på en kjent slik metode er *Nelder-Mead simplex metode*. Metoder som denne kan være nyttige når det ikke er mulig å beregne de deriverte til f, men metodene konvergerer som regel så sakte at de sjelden foretrekkes framfor andre raskere metoder.

[1] Dette kapittelet er basert på bøkene [11], [54] og [42] og notatet "An introduction to numerical methods for unconstrained optimization"av Ole Christian Lingjærde[33]

5.2 Generelle nedstigningsalgoritmer

Generelle nedstigningsalgoritmer benytter seg av informasjon om de deriverte til f i hver iterasjon. Disse metodene krever altså at den førstederiverte, og i noen tilfeller også den andrederiverte, eksisterer og er kontinuerlige. Hvis dette er situasjonen vil som regel disse metodene foretrekkes framfor direkte søkemetoder, ettersom de gir raskere konvergens så lenge ikke utregningene av de deriverte er tidkrevende.

Definer $\nabla f(\mathbf{x}_k)$ som gradientvektoren til f i punktet $\mathbf{x}_k$. Navnet *nedstigning(descent)* kommer av at disse algoritmene for hver iterasjon finner en vektor $\mathbf{s}_k \in \mathbb{R}^p, \quad \mathbf{s} \neq \mathbf{0}$, som kalles *søkeretningen*. Kravet til $\mathbf{s}_k$ er at den skal være i en utforbakke fra $f(\mathbf{x}_k)$, det vil si at verdien til f vil minke dersom vi beveger oss i retningen $\mathbf{s}_k$ fra $\mathbf{x}_k$. Dette kravet tilfredsstilles hvis og bare hvis $\nabla f(\mathbf{x}_k)\mathbf{s}_k < 0$. Algoritme 1 viser pseudokode for en generell nedstigningsalgoritme. Etter man har funnet søkeretningen $\mathbf{s}_k$ må man bestemme hvor langt man skal gå i denne

Algorithm 1 Generell nedstigningsalgoritme

$\mathbf{x}_0 = init$ {et initialt estimat for løsningen $\mathbf{x}^*$}
$k = 0$
while $\mathbf{x}_k$ ikke tilfredsstiller stoppekriterium **do**
  Beregn $\mathbf{s}_k$ slik at $\nabla f(x_k)^T \mathbf{s}_k < 0$
  Beregn α_k slik at $f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) < f(\mathbf{x}_k)$
  $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k$
  $k = k + 1$
end while
return $\mathbf{x}^* = \mathbf{x}_k$

retningen, for å kunne finne neste approksimasjon til løsningen $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k$. Parameteren α_k kalles *skrittlengden*, og den kan finnes ved såkalte *linjesøkingsalgoritmer*. Slike algoritmer vil vi se nærmere på i avsnitt 5.4. Først vil tre metoder for å finne søkeretningen $\mathbf{s}_k$ bli gjennomgått.

En av de enkleste metodene for å gjøre dette kalles *bratteste utforbakke(steepest descent)*. Den velger den negative gradientretningen som søkeretning, $\mathbf{s}_k = -\nabla f(\mathbf{x}_k)$, noe som alltid gir $\nabla f(\mathbf{x}_k)^T \mathbf{s}_k < 0$, dersom $\nabla f(\mathbf{x}_k) \neq \mathbf{0}$. Etter definisjonen av gradientvektoren blir dette blir alltid i den retningen f har brattest utforbakke. Denne metoden konvergerer sent, spesielt dersom konturene til funksjonen er avlange, og blir av denne grunn ofte ansett som uegnet når det kommer til å løse reelle problemer.

Konjugerte metoder benytter seg både av funksjons- og gradientverdiene til f i hver iterasjon til å finne søkeretninger $\mathbf{s}_k, \quad k = 1, \ldots, n$, som er A-konjugerte. Dette vil si at $\mathbf{s}_i A \mathbf{s}_j = 0$ for $i \neq j$, for en positiv definitt matrise A. Denne metoden er effektiv for positiv definitte systemer. (Se [42] for detaljer rundt denne metoden.)

Den siste av metodene som skal nevnes her er *Newtons metode*. Denne metoden bruker informasjon om både den første- og den andrederiverte til f i hver iterasjon. Definer $G_k = \nabla^2 f(\mathbf{x}_k)$, det vil si den som annenderiverte eller *hessematrisen* til f i $\mathbf{x}_k$. Metoden finner søkeretningen som $\mathbf{s}_k = -G_k^{-1} \nabla f(\mathbf{x}_k)$. Dersom G_k er positiv definitt vil dette sikre at vi har en utforbakke

ettersom man får

$$\nabla f(\mathbf{x}_k)^T \mathbf{s}_k = -\nabla f(\mathbf{x}_k)^T G_k^{-1} \nabla f(\mathbf{x}_k) < 0.$$

Noen av ulempene med denne metoden er at man for hver iterasjon må beregne både G_k og så den inverse G_k^{-1}, noe som tar både plass og tid. Dersom G_k blir singulær vil hele metoden bryte sammen. Det eksisterer imidlertid en rekke ulike modifiseringer av denne metoden som unngår slike problemer, og sikrer at den konvergerer raskt og sikkert. *Kvasi-Newton algoritmer* er eksempler på dette. De prøver å unngå problemene rundt beregninger av G_k ved å konstruere en iterativ approksimering H_k^{-1} til G_k^{-1}. Man setter $H_{k+1}^{-1} = H_k^{-1} + C_k$, der C_k beregnes ved å bruke funksjonsverdien, $f(\mathbf{x}_k)$, og gradientverdien, $\nabla f(\mathbf{x}_k)$, til å bygge opp nok krumningsinformasjon til å kunne lage en approksimasjon H_k til hessematrisen, ved hjelp av en passende oppdateringsteknikk. Metodene sikrer at H_k alltid eksisterer og blir symmetrisk og positiv definitt.

Vi skal nå se nærmere på en slik kvasi-Newton metode som bruker en meget effektiv oppdatering av hessematrisen, før vi skal se videre på linjesøkingsalgoritmer for valg av skrittlengden α_k.

5.3 Newtons optimaliseringsalgoritme med BFGS-oppdatering

Vi skal nå se på en variant av Newtons metode for ikke-lineære likninger. Metoden går under navnet *Broyden-Fletcher-Goldfarb-Shanno(BFGS)*. I denne metoden finnes søkeretningen $\mathbf{s}_k$ ved å løse følgende likningssystem for hver hovediterasjon

$$\mathbf{s}_k = -H_k^{-1} \nabla f(\mathbf{x}_k),$$

der H_k^{-1} er en approksimasjon til den inverse av hessematrisen til f i $\mathbf{x}_k$. Denne approksimeringsteknikken for G_k kalles *BFGS-oppdatering*. Oppdateringen går ut på at man for hver iterasjon finner den neste iterasjonens tilnærming H_{k+1}^{-1} ved å bruke formelen

$$H_{k+1}^{-1} = H_k^{-1} + \left(1 + \frac{\mathbf{g}_k^T H_k^{-1} \mathbf{g}_k}{\mathbf{g}_k^T \mathbf{r}_k}\right) \frac{\mathbf{r}_k \mathbf{r}_k^T}{\mathbf{r}_k^T \mathbf{g}_k} - \frac{\mathbf{r}_k \mathbf{g}_k^T H_k^{-1} + H_k^{-1} \mathbf{g}_k \mathbf{r}_k^T}{\mathbf{g}_k^T \mathbf{r}_k},$$

der $\mathbf{g}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$ og $\mathbf{r}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$. Man starter med en passende H_0 (ofte velges identitetsmatrisen I), og beregner så H_{k+1}^{-1} i hver iterasjon. Etter mange års erfaring er det enighet om at BFGS er en metode som har gode egenskaper og gir effektive løsninger på generelle problemer[42, 33].

5.4 Linjesøk

Etter man har funnet søkeretningen, $\mathbf{s}_k$, må man finne en skrittlengde, α_k, som gir en akseptabel $\mathbf{x}_{k+1}$, dvs. vi må finne en α_k som gjør at $\mathbf{x}_{k+1}$ tilfredsstiller visse kriterier, slik at metoden til slutt konvergerer [11]. For at $\mathbf{x}_{k+1}$ skal være akseptabel vil det være naturlig å kreve at $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$, med dette kravet alene sikrer ingen konvergens. Man bør også kreve at avstanden mellom $f(\mathbf{x}_k)$ og $f(\mathbf{x}_{k+1})$ er minst en gitt andel av den initielle avstanden

i den retningen man skal gå. Dette vil si at man velger en parameter $\omega \in (0, 1]$ og velger så en α_k som tilfredsstiller

$$f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) \leq f(\mathbf{x}_k) + \omega\alpha_k \ \nabla f(\mathbf{x}_k)^T \mathbf{s}_k.$$

I tillegg må man forsikre seg om at man ikke tar for små steg. Dette kan gjøres ved å kreve at raten f avtar i retningen $\mathbf{s}_k$ i punktet $\mathbf{x}_{k+1}$ er større enn en gitt andel av raten som f avtar med i retningen $\mathbf{s}_k$ i punktet $\mathbf{x}_k$. Dette kan skrives som

$$\nabla f(\mathbf{x}_k + \alpha_k \mathbf{s}_k)^T \mathbf{s}_k \leq \beta \nabla f(\mathbf{x}_k)^T \mathbf{s}_k,$$

for en gitt konstant $\beta \in (\omega, 1)$. Ved å kreve at $\beta > \omega$ vil begge kravene tilfredsstilles samtidig.

Et linjesøk har som hensikt å finne α_k og kan utføres ved å finne minimumet langs linjen $\mathbf{s}_k$. Dette minimumet kan beregnes som en lokal minimator av $f(\mathbf{x}_x + \alpha\mathbf{s}_k)$, ansett som en funksjon av α. Vi finner da α_k ved å løse problemet

$$\min_{\alpha} f(\mathbf{x}_x + \alpha\mathbf{s}_k). \tag{5.1}$$

Dersom vi definerer funksjonen $\phi : \mathbb{R}^1 \to \mathbb{R}^1$ som $\phi(\alpha) = f(\mathbf{x}_k + \alpha\mathbf{s}_k)$ blir dette tilsvarende med å finne $\phi'(\alpha) = 0$, eller nær nok 0 ettersom hvor strenge kriterier man har. $\phi(\alpha)$ blir ikke lineær, så det kreves en numerisk metode for å estimere α_k, og det er dette som kalles linjesøk. Disse er generelt kostbare beregningsmessig, i og med at de som vi snart kan se kan kreve mange evalueringer av f og ∇f. Effektiviteten til linjesøket er derfor med på å bestemme effektiviteten til hele optimeringsalgoritmen, og det er lagt stor innsats i å lage effektive algoritmer for linjesøk.

En vanlig tilnærming til å finne en løsning på problemet i (5.1) er å bruke en metode som involverer polynomisk interpolasjon eller ekstrapolasjon. Den polynomiske metoden lager iterative approksimasjoner til det univariate polynomet $\phi(\alpha) = f(\mathbf{x}_x + \alpha\mathbf{s}_k)$, og finner ved hjelp av dette minimumet α_k for $\phi(\alpha)$. Slike polynomiske metoder er generelt regnet som effektive dersom funksjonen er kontinuerlig.

Metoden som ble benyttet i denne oppgaven er en slik kvadratisk tilpasningsprosedyre. Prosedyren går ut på å approksimere funksjonen $\phi(\alpha)$ ved et kvadratisk polynom $\Phi(\alpha)$. Dette kan gjøres ved å finne funksjonsverdien til $\phi(\alpha)$ ved tre ulike punkter $\alpha_1, \alpha_2, \alpha_3$, der $\alpha_1 < \alpha_2 < \alpha_3$, slik at $\phi(\alpha_1) < \phi(\alpha_2)$ og $\phi(\alpha_1) < \phi(\alpha_3)$. Da finnes det ved Langrange interpolasjon et unikt kvadratisk polynom $\Phi(\alpha)$ gitt ved likningen:

$$\Phi(\alpha) = \frac{(\alpha - \alpha_2)(\alpha - \alpha_3)}{(\alpha_1 - \alpha_2)(\alpha_1 - \alpha_3)}\phi_1 + \frac{(\alpha - \alpha_1)(\alpha - \alpha_3)}{(\alpha_2 - \alpha_1)(\alpha_2 - \alpha_3)}\phi_2 + \frac{(\alpha - \alpha_1)(\alpha - \alpha_2)}{(\alpha_3 - \alpha_1)(\alpha_3 - \alpha_2)}\phi_3, \tag{5.2}$$

der $\phi_i = \phi(\alpha_i)$ og som gir at $\Phi(\alpha_i) = \phi(\alpha_i)$ for $i = 1, 2, 3$. Dette polynomet kan minimeres ved å derivere (5.2) med hensyn på α og sette den deriverte lik 0. Det kritiske punktet er da gitt ved

$$\alpha^* = \frac{1}{2}\frac{b_{23}\phi_1 + b_{31}\phi_2 + b_{12}\phi_3}{a_{23}\phi_1 + a_{31}\phi_2 + a_{12}\phi_3},$$

der $a_{ij} = \alpha_i - \alpha_j$ og $b_{ij} = \alpha_i^2 - \alpha_j^2$.

Et slikt linjesøk har altså to faser, først må man finne punktene $\alpha_1, \alpha_2, \alpha_3$, som beskrevet over. Dette kalles *innhegningsfasen*. I denne oppgaven ble det brukt et enkelt gridsøk i innhegningsfasen. Algoritme 2 viser hvordan dette ble gjort. Etter dette følger *interpolerings- og*

Algorithm 2 Gridsøk for innhegningsfase

$\alpha_1 = 0$
$\phi_2 = \phi_1 = \phi(0)$
$\alpha_i = \delta = 10^{-4}$ {Gridstørrelse som velges til en liten verdi}
while $\alpha_i < 1$ **do**
  $\phi_i = \phi(\alpha_i)$
  if $\phi_i < \phi_2$ **then**
    $\phi_2 = \phi_i$
    $\alpha_2 = \alpha_i$
  else
    $\phi_3 = \phi_i$
    $\alpha_3 = \alpha_i$
    return
  end if
  $\alpha_i = \alpha_i + \delta$
end while

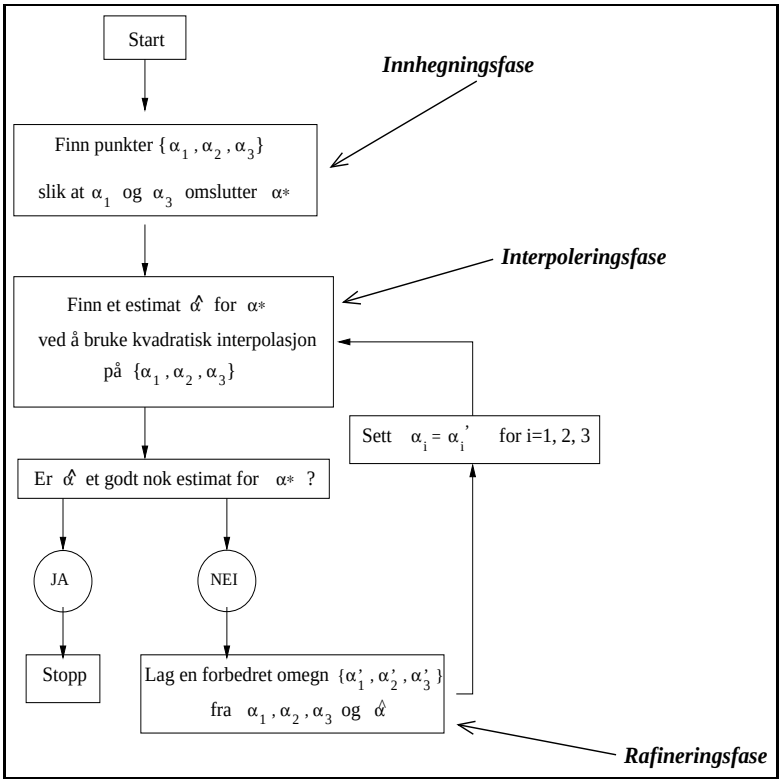
rafineringsfasen der man gjentatte ganger gjør en interpolering for nye punkter $\alpha_1', \alpha_2', \alpha_3'$ helt til man finner en $\hat{\alpha}$ som gjør at $\phi'(\hat{\alpha}) \approx 0$. Denne prosessen kan illustreres ved flytdiagram som i figur 5.1.

5.5 En mer effektiv algoritme

I kapittel 4.5 så vi at det å løse et lineært ridge-regresjonsproblem med p kovariater innebærer å invertere en $p \times p$ matrise $(X^T X + \lambda I)$. Dette krever $O(p^3)$ operasjoner. Tidsbruken for å finne et estimat for $\boldsymbol{\beta}$ for en ordinær lineær modell vil være tilsvarende. Hastie og Tibshirani foreslo i 2003[19] en måte å effektivisere slike beregninger når man har en kovariatmatrise $X \in \mathbb{R}^{n \times p}$ der $n << p$, slik man typisk har med mikroarraydata. De presenterer teknikker for å gjøre en slik effektivisering både for klassisk lineær ridge regresjon og ridge regresjon for generelle lineære modeller, det vil si modeller der kovariatene inngår som en lineær funksjon. Den straffede Cox-modellen fra (4.20) er et eksempel på en slik modell. Metoden de foreslo går ut på å benytte en QR-faktorisering av X for å redusere dimensjonaliteten i optimeringsproblemet. Vi skal gå gjennom hovedideene bak den foreslåtte metoden og se hvordan dette kan brukes for å effektivisere tilpasningen av den straffede Cox modellen. Vi starter med noen nyttige definisjoner før vi skal se hvordan disse kan brukes til å redusere dimensjonen til optimeringsproblemet.

5.5.1 QR-faktorisering

QR-faktorisering er en nyttig matrisefaktorisering som er mye brukt innen numerisk lineær algebra. Ved å bruke en slik faktorisering er man garantert numerisk stabilitet ved at man minimerer avrundingsfeil som skyldes maskinavrunding. *QR-faktoriseringen* til en matrise kan defineres som følgende:



Figur 5.1: Flytdiagram for å finne minimum ved hjelp av iterativ kvadratisk interpolasjon

- Gitt en matrise $A \in \mathbb{R}^{p\times n}$ av full rang n, og $p \geq n$, da kan A faktoriseres i et produkt $A = QR$, der $Q \in \mathbb{R}^{p\times p}$ har ortonormale kolonner (dvs. $Q^TQ = I$) og $R \in \mathbb{R}^{p\times n}$ er øvre triangulær, dvs. elementene $r_{ij} \geq 0$ for $i \leq j$ og $r_{ij} = 0$ for $i > j$.

Legg merke til at QR-faktoriseringen her defineres for en matrise med samme dimensjon som X^T, den transponerte av kovariatmatrisen i fra et genekspresjonsdatasett. Hensikten med dette kommer fram senere i avsnittet. Dersom man er i en situasjon der $p > n$, kan man se ut fra definisjonen over at de $p - n$ radene i matrisen R bare vil inneholde nuller, ettersom den er øvre triangulær. Dette kan utnytts til å finne en kompakt QR-faktorisering av en slik matrise. En slik *redusert QR-faktorisering* kan defineres som følgende:

- Dersom $A \in \mathbb{R}^{p\times n}$ og $p > n$ har A en redusert QR faktorisering $A = Q_1R_1$, der $Q_1 \in \mathbb{R}^{p\times n}$ har ortonormale kolonner, og $R \in \mathbb{R}^{n\times n}$ er øvre triangulær. Kolonnene i Q_1 utspenner da kolonnerommet til A.

Beviser for begge disse definisjonene kan blant annet finnes i [30].

Et av målene med denne oppgaven er å kunne tilpasse en Cox-modell til et mikroarraydatasett med mange gener, og få observerte individer. Det er mulig å tilpasse en slik modell ved å bruke optimeringsalgoritmene beskrevet i forrige avsnitt, men tidsbruken til denne algoritmen øker betraktelig med antall kovariater, det vil si gener, som er med i estimeringen. Det er ikke ønskelig at effektiviteten til optimeringsprosedyren skal begrense antall gener som kan taes

med i modellen. Metoden Hastie og Tibshirani foreslår er en enkel men effektiv måte å unngå slike problemer på ved å benytte seg av QR-faktorisering når man skal tilpasse en generell lineær modell med ridge straff. Vi starter med å se på hvordan dette kan gjøres for en enkel lineær modell som vi så på i avsnitt 4.2. Deretter skal vi se at de samme ideene også kan overføres til generelle lineære modeller.

5.5.2 Effektivisering av lineært problem

Vi husker fra 4.2 at man kan tilpasse en enkel lineær modell med ridgestraff ved å finne løsningen fra (4.13). Legg merke til at dette blant annet innebærer å regne ut $(X^TX)^{-1}$, noe som tar lang tid når p er stor. Anta at QR-faktoriseringen $X^T = QR$ er funnet og at dette settes inn i løsningen for ridge problemet 4.13. Etter litt omregning får man da

$$\hat{\boldsymbol{\beta}}(\lambda) = Q_1(R_1R_1^T + \lambda I)^{-1}R_1\mathbf{y}, \tag{5.3}$$

der $X^T = Q_1R_1$ er den reduserte QR-faktoriseringen til X^T. (Denne omregningen vil bli vist etter at vi har sett nærmere på hvordan dette reduserer dimensjonaliteten til problemet.)

Løsningen (5.3) likner løsningen til standard ridge regresjon. Ved å sette

$$\tilde{\boldsymbol{\beta}}(\lambda) = (R_1R_1^T + \lambda I)^{-1}R\mathbf{y}, \tag{5.4}$$

finner man løsningen $\tilde{\boldsymbol{\beta}}_{RR} \in \mathbb{R}^n$ bare ved å bruke den n-dimensjonale matrisen R_1. Fra (5.3) kan man finne $\hat{\boldsymbol{\beta}}_{RR}$ enkelt ved å premultiplisere denne løsningen med Q_1^T, det vil si at man finner løsningen på det fulle problemet ved å sette $\hat{\boldsymbol{\beta}}_{RR} = Q_1^T\tilde{\boldsymbol{\beta}}_{RR}$.

▶ **Bevis** Vi skal nå se hvordan man kommer fram til (5.3). Dersom man setter inn for $X^T = QR$ i (4.13) får vi

$$\hat{\boldsymbol{\beta}}_{RR} = (QRR^TQ^T + \lambda I)^{-1}QR\mathbf{y}$$

Premultipliserer med Q^{-1} på hver side og flytter den inn i parentesen på høyre side

$$\begin{aligned} Q^{-1}\hat{\boldsymbol{\beta}}_{RR} &= (QRR^TQ^TQ + \lambda Q)^{-1}QR\mathbf{y} = \\ &= (RR^T + \lambda I)^{-1}Q^{-1}QR\mathbf{y}. \end{aligned}$$

Løser man dette for $\hat{\boldsymbol{\beta}}_{RR}$ får man

$$\hat{\boldsymbol{\beta}}_{RR} = Q(RR^T + \lambda I)^{-1}R\mathbf{y}.$$

Både Q og R kan blokkpartisjoneres ved å sette $Q = [Q_1Q_2]$ og $R = \binom{R_1}{0}$. Disse blokkpar-

tisjoneringene av Q og R settes inn i uttrykket for $\hat{\boldsymbol{\beta}}_{RR}$ og dette blir da:

$$\begin{aligned}
\hat{\boldsymbol{\beta}}_{RR} &= [Q_1 Q_2] \left(\begin{bmatrix} R_1 \\ 0 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix}^T + \lambda I \right)^{-1} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \mathbf{y} = \\
&= [Q_1 Q_2] \left(\begin{bmatrix} R_1 R_1^T & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} \lambda I & 0 \\ 0 & \lambda I \end{bmatrix} \right)^{-1} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \mathbf{y} = \\
&= [Q_1 Q_2] \begin{pmatrix} \left(R_1 R_1^T + \lambda I\right)^{-1} & 0 \\ 0 & 0 \end{pmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \mathbf{y} = \\
&= [Q_1 Q_2] \begin{pmatrix} \left(R_1 R_1^T + \lambda I\right)^{-1} R_1 \\ 0 \end{pmatrix} \mathbf{y} = \\
&= Q_1 \left(R_1 R_1^T + \lambda I\right)^{-1} R_1 \mathbf{y},
\end{aligned}$$

som er den løsningen vi så i (5.4).
□

Dersom vi oppsummerer dette ser vi at med et ridge-problem der $X \in \mathbb{R}^{n \times p}$ og $p > n$ kan man få redusert totalt antall operasjoner det krever å beregne den p-dimensjonale løsningen til problemet ved å først løse et n-dimensjonalt problem, for så å vende tilbake til den p-dimensjonale løsningen ved en enkel matrisemultiplikasjon. Følgende steg kan altså gjøres for å redusere antall operasjoner:

- Finn den reduserte QR-dekomposisjonen $X^T = Q_1 R_1$. Dette krever $O(pn^2)$ operasjoner.
- Løs det n-dimensjonale ridge regresjonsproblemet (5.4) for å finne $\tilde{\boldsymbol{\beta}}_{RR}$. Dette krever $O(n^3)$ operasjoner.
- Transformere løsningen tilbake til p dimensjoner ved å regne ut $\hat{\boldsymbol{\beta}}_{RR} = Q_1^T \tilde{\boldsymbol{\beta}}_{RR}$. Dette krever $O(pn)$ operasjoner.

Dersom $p > n$ kan man altså redusere størrelsesordenen til beregningene fra $O(p^3)$ til $O(pn^2)$ operasjoner. Spesielt vil dette være nyttig når $p >> n$, ettersom gevinsten ved å gjøre da dette vil være stor. Dersom p ikke er så mye større enn n vil man tjene mindre på å bruke en slik metode, ettersom også det å finne QR-faktoriseringen tar tid.

5.5.3 Effektivisering av generelt lineært problem

Hastie og Tibshirani presenterer videre en tilsvarende effektivisering for en generell lineær modell med ridge straff ved hjelp av QR-faktorisering. En løsning på et slikt problem kan finnes ved det generelle prinsippet for ML-estimering. Ved hjelp av QR-faktoriseringen vil man kunne redusere antall operasjoner en slik optimering krever, på tilsvarende måte som i det lineære tilfellet.

Teorem 5.5.1 (Effektivisering av ridge regresjon for generelle lineære modeller). *Anta at en $X \in \mathbb{R}^{n \times p}$ og en lossfunksjon, L er gitt. La $X^T = QR$ være QR-faktoriseringen,*

og $X^T = Q_1 R_1$ *være den reduserte QR-faktoriseringen av* X^T*. La videre*

$$L_1(\beta_0, \boldsymbol{\beta}) = \sum_{i=1}^{n} L(y_i, \beta_0 + \mathbf{x}_i^T \boldsymbol{\beta}) - \lambda \boldsymbol{\beta}^T \boldsymbol{\beta} \tag{5.5}$$

og

$$L_2(\beta_0, \tilde{\boldsymbol{\beta}}) = \sum_{i=1}^{n} L(y_i, \beta_0 + \mathbf{r}_i^T \tilde{\boldsymbol{\beta}})) - \lambda \tilde{\boldsymbol{\beta}}^T \tilde{\boldsymbol{\beta}} \tag{5.6}$$

der $\mathbf{r}_i^T = \mathbf{x}_i^T Q_1$*. La så* $(\hat{\beta}_0, \hat{\boldsymbol{\beta}})$ *være et minimum for* L_2*. Da er* $(\hat{\beta}_0, Q_1 \hat{\boldsymbol{\beta}})$ *et minimum for* L_1*.*

▶ **Bevis** Beviset for dette teoremet kan finnes i [19]. (med en litt annen notasjon.)
□

Det å finne $\hat{\boldsymbol{\beta}}$ ved å bruke løsningen skissert i teorem 5.5.1 vil gi en tilsvarende reduksjon i antall operasjoner som i det lineære tilfellet i forrige avsnitt. I Cox-modellen inngår X bare i en lineær sammenheng, så denne metoden for å redusere dimensjonen til problemet vil kunne benyttes på problemet i denne oppgaven. Ettersom vi ser på mikroarraydata der det som regel er et stort antall parametere og få observasjoner, kan vi forvente at tiden det tar å estimere den straffede Cox-modellen ved bruk av denne metoden vil reduseres vesentlig i forhold til om man optimaliserte ved å bruke den ordinære metoden.

Kapittel 6

Modellseleksjon

6.1 Innledning

Fram til nå har vi antatt at ridgeparameteren λ er satt til en fast verdi når likelihooden optimeres. I en praktisk situasjon vil λ være en ukjent parameter som må bestemmes på linje med andre parametere i modellen. En framgangsmåte for å bestemme parameteren vil være å estimere hvilken verdi av λ som er best etter et gitt kriterium. Et forslag ville være å finne en λ som maksimerer likelihooden, men vi skal se at ikke dette er noen god ide. Anta at vi har en straffet log-likelihood $F(\boldsymbol{\beta}, \lambda) = -l(\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta}$, der $l(\boldsymbol{\beta})$ er den ustraffede log-likelihooden. For en gitt $\lambda \geq 0$ kan man skrive om ML-løsningen til et slikt problem som

$$\min_{\boldsymbol{\beta}} \{-l(\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta}\} \iff \min_{\boldsymbol{\beta}} \{-l(\boldsymbol{\beta})\} \text{ slik at } \boldsymbol{\beta}^T \boldsymbol{\beta} \leq C_\lambda,$$

altså ved å legge et eksplisitt krav på størrelsen til parameterne, slik at det er en en-til-en sammenheng mellom λ og C_λ[21]. Vi prøver så å finne både ML-estimater for $\boldsymbol{\beta}$ og λ ved å maksimere $F(\boldsymbol{\beta}, \lambda)$. Ved å bruke denne omskrivingen kan dette problemet splittes opp slik at vi får:

$$\min_{\boldsymbol{\beta}, \lambda} F(\boldsymbol{\beta}, \lambda) = \min_{\lambda} \left\{ \min_{\boldsymbol{\beta}} F(\boldsymbol{\beta}, \lambda) \right\} = \min_{\lambda} F(\hat{\boldsymbol{\beta}}_\lambda, \lambda),$$

der $\hat{\boldsymbol{\beta}}_\lambda$ er et uttrykk for ML-estimatet av $\boldsymbol{\beta}$ som avhenger av verdien λ. Vi vet også at $\hat{\boldsymbol{\beta}}_0$ er ML-estimatet når $\lambda = 0$, det vil si når man ikke har noen krav til størrelsen på parameterne. Vi kan da skrive dette uttrykket som:

$$\begin{aligned} \min_{\lambda} F(\hat{\boldsymbol{\beta}}_\lambda, \lambda) &= \min_{\lambda} \{-l(\hat{\boldsymbol{\beta}}_\lambda) + \lambda \hat{\boldsymbol{\beta}}_\lambda^T \hat{\boldsymbol{\beta}}_\lambda\} \geq \min_{\lambda} \{-l(\hat{\boldsymbol{\beta}}_0) + \lambda \hat{\boldsymbol{\beta}}_\lambda^T \hat{\boldsymbol{\beta}}_\lambda\} \\ &= -l(\hat{\boldsymbol{\beta}}_0) + \min_{\lambda} \lambda \hat{\boldsymbol{\beta}}(\lambda)^T \hat{\boldsymbol{\beta}}(\lambda) = -l(\hat{\boldsymbol{\beta}}_0) = F(\hat{\boldsymbol{\beta}}_0, 0), \end{aligned}$$

Ulikheten vil holde fordi vi vet at $\hat{\boldsymbol{\beta}}_0$ per definisjon er det estimatet som minimerer den negative likelihooden. Som vi ser vil det å finne $\lambda \geq 0$ ved å minimere den straffede log-likelihooden $F(\boldsymbol{\beta}, \lambda)$ alltid resultere i $\lambda = 0$, det vil si ingen straff. Dette kan sees på som en form for overtilpasning, ettersom det resulterer i at man alltid vil velge den ustraffede modellen. Som nevnt tidligere kan denne være svært følsom for perturbasjoner i dataene. Et

av poengene med å regularisere var nettopp å unngå slike problemer, så denne tilnærmingen til å bestemme λ vil være meningsløs.

Vi skal i dette kapittelet se på andre tilnærminger til modellseleksjon. Kapittelet er delt i 3 hoveddeler. Vi starter med å gå gjennom prinsippene for kryssvalidering. Deretter vil tre forslag for hvordan dette kan gjøres for Cox partielle likelihood vil bli presentert og sammenliknet. Til slutt skal vi se kort på en alternativ modellseleksjonsmetode kalt L-kurven.

6.2 Kryssvalidering for ridge regresjon

Kryssvalidering er kanskje den enkleste og mest brukte metoden for modellseleksjon. Metoden går kort sagt ut på at man fjerner en og en observasjon fra datasettet, for så å tilpasse modellen på bakgrunn av de $n-1$ resterende observasjonene. Man bruker så det utelatte individet som et slags testsett ved å beregne hvor godt dette individet "passer" med den estimerte modellen. Vi skal først se på en grunnleggende form for kryssvalidering. Prinsippet for kryssvalidering vil ta utgangspunkt i en enkel lineær regresjonsmodell med ridge straff. I avsnitt 6.4 vil vi ta for oss kryssvalidering i en mer generell situasjon, der en straffet likelihood skal optimeres.

En estimert ordinær lineær modell med ridge-straff kan skrives som (se avsnitt 4.6):

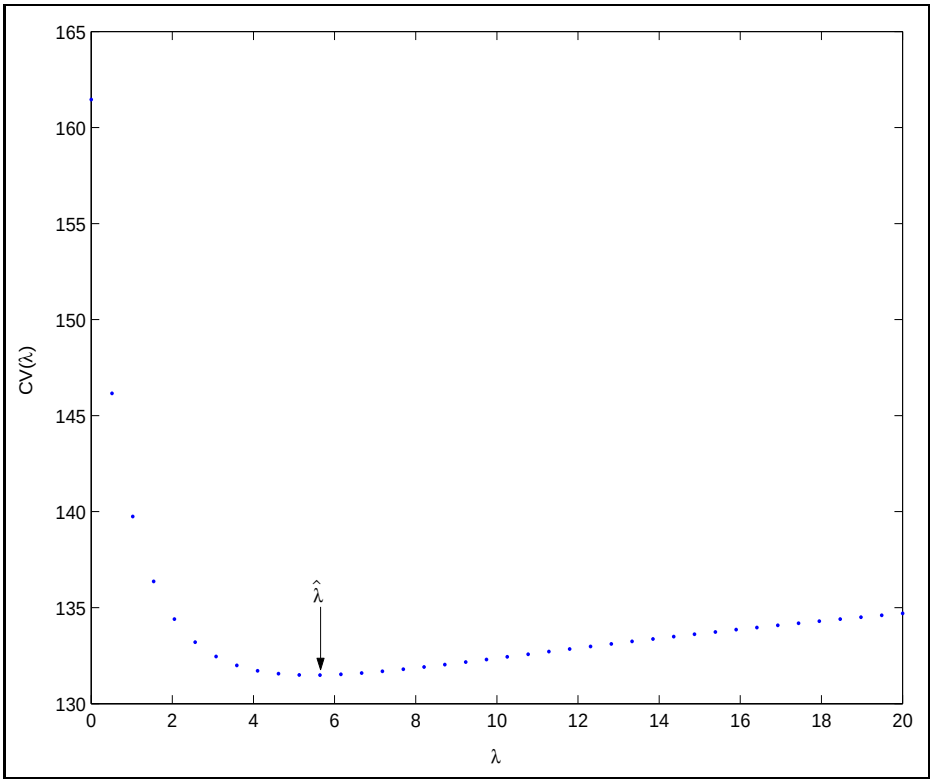
$$\hat{f}_\lambda(\mathbf{x}_i) = \hat{\boldsymbol{\beta}}(\lambda)^T \mathbf{x}_i,$$

der y_i er responsen og $\mathbf{x}_i$ er vektoren av de målte kovariatene for observasjonene $i = 1, \ldots, n$, mens $\lambda \geq 0$ er straffeparameteren. En slik modell estimeres ved å finne den $\hat{\boldsymbol{\beta}}(\lambda)$ som minimerer uttrykket $\sum_{i=1}^n \{y_i - \boldsymbol{\beta}^T \mathbf{x}_i\}^2 + \lambda \sum_{j=1}^p \beta_j$.

For å kunne teste prediksjonsfeilen til en slik modell kan man bruke kryssvalidering. Dette gjøres ved at man fjerner en observasjon fra datasettet, og finner et estimat $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ som over, men på bakgrunn av det forkortede datasettet. Den opphøyde indeksen $-i$ vil si at estimatet er basert på datasettet uten observasjon i. Et slik estimat, beregnet på bakgrunn av et forkortet datasett, kalles et *jackknife-estimat*[12]. Dette kan brukes til å finne hvor godt den estimerte modellen $\hat{f}_\lambda^{-i}(\mathbf{x}) = (\hat{\boldsymbol{\beta}}(\lambda)^{-i})^T \mathbf{x}$ klarer å predikere responsen y_i til den utelatte observasjonen med kovariater $\mathbf{x}_i$, ved å se på prediksjonsfeilen $\{y_i - \hat{f}_\lambda^{-i}(\mathbf{x}_i)\}^2$. Denne kvadrerte feilen kan sees på som den i'te observasjonens bidrag til modellen, det er et mål på hvor godt den i'te observasjonen passer modellen. Alle observasjonene i datasettet taes ut en etter en på denne måten, og prediksjonsfeilen beregnes for hver av dem. Deretter finner man gjennomsnittet av prediksjonsfeilene for alle observasjonene. Dette gir oss kryssvalideringsmålet

$$CV(\lambda|data) = \frac{1}{n} \sum_{i=1}^n \{y_i - \hat{f}_\lambda^{-i}(\mathbf{x}_i)\}^2, \tag{6.1}$$

der *data* er alle observasjonene vi har, dvs. dataparene $(\mathbf{x}_i, y_i)$ for $i = 1, \ldots, n$. Denne funksjonen blir et estimat for sann prediksjonsfeil for ett valg av ridgeparameteren λ. Dersom man beregner $CV(\lambda|data)$ for ulike verdier av λ, si $\lambda = \lambda_k$ for $k = 1, \ldots, D$, og plotter disse får man det som kalles et *kryssvalideringsplott*. Man kan finne den λ_k som minimerer denne kurven, det vil si minimerer $CV(\lambda|data)$, og sette $\hat{\lambda} = \lambda_k$. Det er ofte ønskelig med stor D, ettersom man da får en bedre oppløsning for kryssvalideringsplottet, og det vil dermed bli en



Figur 6.1: Et eksempel på et kryssvalideringsplott med 40 ulike verdier $\lambda_k, k = 1, \ldots, 40$ for λ. Den optimale $\hat{\lambda}$ er merket av som den verdien λ_k som minimerer funksjonen $CV(\lambda)$.

mer nøyaktig vurdering av hvilken verdi den optimale λ har. Et eksempel på et kryssvalideringsplott er vist i figur 6.1.

Et minimum til funksjonen $CV(\lambda|data)$ kan også finnes ved å bruke mer effektive metoder enn å benytte et uniformt grid. Man kan for eksempel velge punktene λ_k dynamisk ved å bruke en metode kalt *golden section search*[42]. Denne går ut på at man starter med tre verdier $\lambda_a < \lambda_b < \lambda_c$ som er slik at $CV(\lambda_a) > CV(\lambda_b) < CV(\lambda_c)$. Man velger så et punkt λ_x enten mellom λ_a og λ_b eller mellom λ_b og λ_c. La oss for eksempel si at vi velger det siste. Dersom $CV(\lambda_b) < CV(\lambda_x)$ blir de nye punktene $(\lambda_a, \lambda_b, \lambda_x)$, i motsatt fall dersom $CV(\lambda_b) > CV(\lambda_x)$ blir de nye punktene $(\lambda_b, \lambda_x, \lambda_c)$. Midpunktet blir alltid det minste punktet man har sett så langt. Deretter velger man et nytt punkt λ_x og fortsetter til avstanden mellom de to ytterpunktene er liten nok. Til slutt setter man $\hat{\lambda}$ lik det minimumet man fant.

Denne metoden vil generelt være raskere til å finne minimum enn grid-metoden vi så på over. Ulempen med å velge en slik søkemetode for å finne $\hat{\lambda}$ er at man ikke vet om funksjonen $CV(\lambda|data)$ er konveks eller om $CV(\lambda|data)$ har et entydig minimum. Ved å plotte funksjonen for et uniformt grid som forklart over vil det kunne være lettere å avdekke slike situasjoner. Hvilken metode som bør benyttes til å finne minimum av $CV(\lambda|data)$ bør vurderes i den enkelte situasjon. Når man har gjennomført en full kryssvalidering og funnet $\hat{\lambda}$ vil den endelige modellen bli $\hat{f}_\lambda(\mathbf{x})$, som er tilpasset med alle observasjonene.

I dette eksempelet var det ridgeparameteren λ som gir den beste modellen vi ønsket å finne. Generelt kalles en slik parameter man justerer under kryssvalideringen for en *tuningparameter*. Kryssvalideringsmetoder kan også utvides til en situasjon der man har en vektor med q ukjente tuningparametere $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_q)$. En slik situasjon har man for eksempel ved generalisert ridge regresjon, som ble nevnt i avsnitt 4.6. Dette innebar å kunne velge å straffe de ulike komponentene i $\boldsymbol{\beta}$ forskjellig, ved å skrive straffeleddet til modellen som $\boldsymbol{\beta}^T Q \boldsymbol{\beta}$, der Q er en diagonalmatrise med verdiene $\lambda_1, \ldots, \lambda_q$ på diagonalen. I en slik situasjon kryssvaliderer man ved å variere alle de q ulike parameterene, slik at man finner den optimale parametervektoren $\hat{\boldsymbol{\lambda}}$.

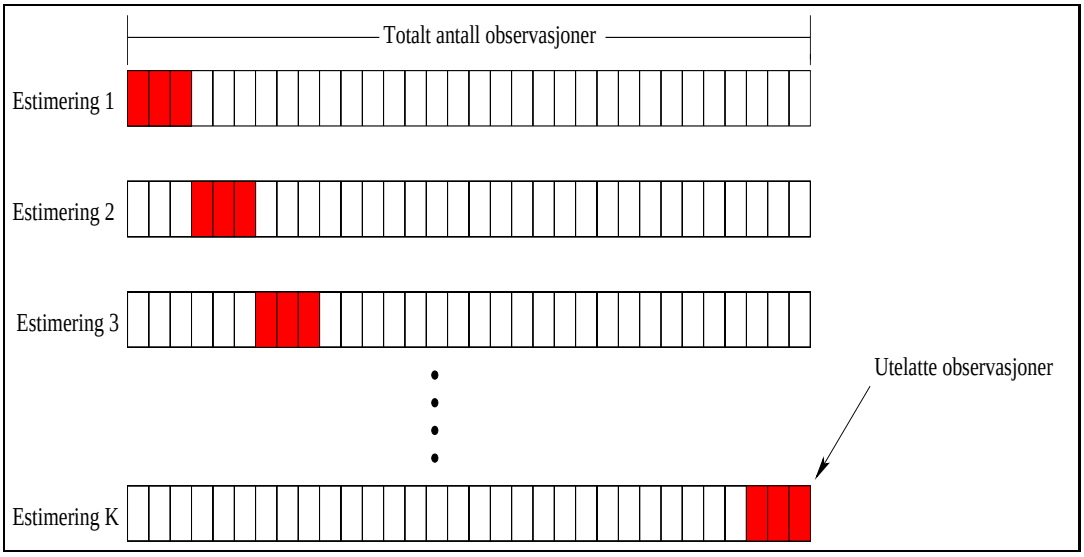
6.3 K-fold kryssvalidering

Kryssvalidering som beskrevet i avsnittet over kan gå veldig tregt dersom man har mange observasjoner og benytter mange gridpunkter λ_k for λ. Et raskere alternativ er å gjøre en k-fold kryssvalidering. Dette er en enkel generalisering av situasjonen i avsnittet over, der man randomiserer datasettet inn i K blokker $I_1, ..., I_K$, av tilnærmet lik størrelse $m = n/K$. Kryssvalideringskriteriet $CV_K(\lambda|data)$ defineres deretter som

$$CV_K(\lambda|data) = \frac{1}{K} \sum_{j=1}^{K} err_j,$$

der $err_j = \frac{1}{m} \sum_{k \in I_j} \{y_k - \hat{f}_\lambda^{-I_j}(\mathbf{x}_k)\}^2$ og $\hat{f}_\lambda^{-I_j}(\mathbf{x})$ er modellen estimert på bakgrunn av de $K-1$ blokkene $I_i,\ i \neq j$. Man tar nå altså ut m individer av gangen. Når $K = n$ blir dette tilsvarende som vi beskrevet over, der ett og ett individ ble fjernet av gangen. Se figur 6.2 for en illustrasjon av k-fold kryssvalidering.

Det finnes ingen fasitsvar på hvordan man bør velge antall blokker K. Dersom man setter $K = n$ er estimatoren for den sanne prediksjonsfeilen (CV) så og si uten bias, ettersom man bare tar ut ett og ett individ. Estimatoren kan derimot få høy varians, fordi man trener på n nesten helt like datasett[21]. Den store ulempen med å velge $K = n$ er som sagt at det kan ta lang tid å gjøre en slik *leave-one-out* kryssvalidering. En av fordelene med å velge $K < n$ er at man får redusert antall modelltilpasninger, og derfor reduseres også kjøretiden. Estimatoren $CV(\lambda|data)$ vil i tillegg kunne få liten varians. Overestimering av prediksjonsfeilen (økt bias) vil derimot kunne bli et problem i slike tilfeller. Dette avhenger av hvor sensitiv modellen som trenes opp er for størrelsen på treningsdatasettet, det vil si de $K-1$ blokkene som brukes i hver estimering av modellen. I praksis bestemmes ofte størrelsen på K etter størrelsen på datasettet. Har man små datasett, er det ofte nødvendig å bruke leave-one-out kryssvalidering, for å trene på så mange eksempler som mulig. Ellers er $K = 5$ eller $K = 10$ vanlig å bruke[21], det vil si at man deler datasettet sitt i henholdsvis 5 eller 10 like deler.



Figur 6.2: Illustrasjon av K-fold kryssvalidering. For hver estimering holder man ute m observasjoner og estimerer modellen på bakgrunn av de resterende observasjonene. Her er $m = 3$ og $n = 18$. Dette resulterer i K ulike estimerte modeller, ettersom hver av de K blokkene må holdes ute en gang. Data er her randomisert med hensyn på overlevelsestid.

6.4 Kryssvalidering for generelle lineære modeller med ridge straff

I en mer generell situasjon enn den vi har sett på til nå, vil man kunne ha en straffet modell som baseres på likelihoodestimering. En log-likelihood med ridge-type straff vil som vi husker fra avsnitt 4.6.3 ha formen :

$$F(\boldsymbol{\beta}, \lambda) = l(\boldsymbol{\beta}|data) - \lambda \boldsymbol{\beta}^T \boldsymbol{\beta},$$

der l er log-likelihooden. Vi ønsker å utvide prinsippene for kryssvalidering som vi så på i de to forrige avsnittene til å kunne finne den beste straffeparameteren λ for en slik modell. Det er ikke noen entydig måte å definere kryssvalideringskriteriet $CV(\lambda|data)$ for slike modeller, ettersom det ikke er klart hva man bør bruke som mål på hvor godt nye observasjoner passer med modellen. De løsningene vil skal se på her har alle formen

$$CV(\lambda|data) = \frac{1}{n} \sum_{i=1}^{n} l_i(\hat{\boldsymbol{\beta}}(\lambda)^{-i}|data), \tag{6.2}$$

der $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ er ML-estimatet som er beregnet fra datasettet der den i'te observasjonen er holdt utenfor og $l_i(\boldsymbol{\beta}|data)$ er det i'te bidraget til log-likelihoodsummen. Vi skal se kort hvorfor dette kan være et rimelig kriterium for kryssvalidering. Anta at $data$ er generelle data, som passer til den likelihoodmodellen vi studerer og at $data^{-i}$ er datasettet der den i'te observasjonen er tatt ut. Dersom hver observasjon gir uavhengige bidrag til log-likelihooden, vil dette innebære at man kan skrive den ustraffede log-likelihooden som en sum av enkelt bidrag fra hver av de n observasjonene, $l(\boldsymbol{\beta}|data) = \sum_{i=1}^{n} l_i(\boldsymbol{\beta}|data)$.

Analogt med ideene fra avsnitt 6.2 kan man da finne et jackknife-estimat $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ ved å maksimere den straffede log-likelihooden $l(\boldsymbol{\beta}|\lambda, data^{-i})$ på bakgrunn av datasettet der man har tatt ut den i'te observasjonen. Man beregner så det i'te bidraget til den ustraffede log-likelihooden som $l_i(\hat{\boldsymbol{\beta}}(\lambda)^{-i}|data)$. Dette gjøres for en og en observasjon, og kryssvalideringskriteriet $CV(\lambda|data)$ som man ønsker å minimere defineres som summen over alle disse bidragene, slik vi så i (6.2).

Bidraget l_i kan tolkes som et mål på hvor godt observasjon i passer med modellen som har blitt estimert. Dersom det blir et stort bidrag vil dette øke log-likelihoodsummen, dvs. at likelihooden øker, og man kan da si at individet passer bra med den estimerte modellen. Man ønsker derfor i dette tilfellet å maksimere $CV(\lambda|data)$ ettersom man ønsker å maksimere likelihooden. Dette blir det motsatte av tilfellet med lineære modeller, der man ønsket å minimere kvadrert feil i kryssvalideringkriteriet (6.1), men prinsippene blir de samme.

K-fold kryssvalidering blir også tilsvarende som generaliseringen for en straffet ordinær lineær modell, bortsett fra at man nå finner den λ som maksimerer

$$CV_K(\lambda|data) = \frac{1}{K}\sum_{j=1}^{K} \tilde{l}_j(\hat{\boldsymbol{\beta}}(\lambda)^{-I_j}|data),$$

der $\hat{\boldsymbol{\beta}}(\lambda)^{-I_j}$ nå er jackknife-estimatet på basis av de $K-1$ blokkene $I_i,\ i \neq j$. Log-likelihoodbidraget $\tilde{l}_j$ blir $\tilde{l}_j(\hat{\boldsymbol{\beta}}(\lambda)^{-I_j}|data) = \sum_{k\in I_j} l_k(\hat{\boldsymbol{\beta}}(\lambda)^{-I_j}|data)$, som tilsvarer summen av bidragene fra de m individene i blokk j.

6.5 Kryssvalidering for Cox partielle likelihood

Vi skal se nærmere på hvordan man kan utføre kryssvalidering for en spesiell straffet modell, nemlig Cox proporsjonale hasardmodell med en ridge-type straff. Denne ble presentert i avsnitt 4.7 og vi så der at den straffede partielle log-likelihooden til Cox-modellen kan skrives som:

$$l_{pen}(\boldsymbol{\beta}, \lambda) = l(\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta} = \sum_{i=1}^{n} \delta_i\{(\boldsymbol{\beta}^T\mathbf{x}_i) - log\{\sum_{j\in\mathcal{R}_{(t_i)}} exp(\boldsymbol{\beta}^T\mathbf{x}_j)\}\} + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}. \tag{6.3}$$

Vi definerer som tidligere for Cox-modellen $data_j$ for individ j, $\quad j = 1, \ldots, n$ som tripletter bestående av tid t_j, sensurindikator δ_j og koeffisientvektor $\mathbf{x}_j$. For å bruke en enklere notasjon videre i kapittelet definerer vi $\varphi_i(\boldsymbol{\beta})$ som det i'te leddet i den ustraffede log-likelihooden til denne modellen, det vil si at

$$\varphi_i(\boldsymbol{\beta}) = \delta_i\{\boldsymbol{\beta}^T\mathbf{x}_i - log\{\sum_{j\in\mathcal{R}_{(t_i)}} exp(\boldsymbol{\beta}^T\mathbf{x}_j)\}\}. \tag{6.4}$$

Vi har da at den straffede likelihooden i (6.3)kan skrives som

$$l_{pen}(\boldsymbol{\beta}, \lambda) = \sum_{i=1}^{n} \varphi_i(\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}.$$

Vi skal nå se på hvordan kryssvalidering kan brukes til å velge mellom ulike estimater av den straffede Cox-modellen, der det som skiller de estimerte modellene er hvilken verdi som er valgt for straffeparameteren λ. Vi så i forrige avsnitt at dersom komponentene i log-likelihoodfunksjonen til en likelihoodmodell var uavhengige, kunne man se på bidraget til den i'te observasjonen som det i'te leddet i denne summen. I den partielle log-likelihooden til Cox-modellen er derimot ikke bidragene uavhengige. Vi kan se fra (6.4) at hvert ledd i den partielle log-likelihoodsummen inneholder en risikogruppe, $\mathcal{R}_{(t_i)}$ som består av informasjon fra en eller flere av observasjonene i datasettet. Dette medfører at observasjon i også kan bidra til andre ledd i log-likelihoodsummen enn bare det i'te. Dette gjør at det ikke er helt rett fram hvordan man utfører en kryssvalidering for den partielle log-likelihooden til Cox-modellen. Det er gjort ulike forsøk på å gjøre dette, og vi skal se på tre ulike tilnærminger her. Den første er etter Kuk [29], den andre er foreslått av Verweij og van Houwelingen [52] mens den siste er et nytt forslag til hvordan en slik kryssvalidering kan gjøres. Til slutt vil det bli gjort et forsøk på å vurdere de ulike metodene ved å sammenlikne dem med hverandre.

6.5.1 Variant 1

Den første varianten av kryssvalidering ble foreslått av Kuk i 1984[29]. Han argumenterer for å kryssvalidere i en proporsjonal hasardmodell ved å endre statusen til en observasjon fra usensurert til sensurert. Han starter med å presentere likelihooden for Cox-modellen:

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{k} \frac{exp(\boldsymbol{\beta}^T \mathbf{x}_{(i)})}{\sum_{j \in R_{(t_{(i)})}} exp(\boldsymbol{\beta}^T \mathbf{x}_j)},$$

der $t_{(1)} < \ldots < t_{(k)}$ er de usensurerte hendelsestidene for individer med tilsvarende kovariatvektorer $\mathbf{x}_{(1)}, \ldots, \mathbf{x}(k)$ og sensurstatuser $\delta_{(1)}, \ldots, \delta_{(k)}$. Det er altså k av de n hendelsestidene som ikke er sensurerte. Kuks kriterium for valg av modell er å finne den straffeparameteren λ som maksimerer

$$CV_1(\lambda|data) = \sum_{i=1}^{k} \varphi_{(i)}(\hat{\boldsymbol{\beta}}(\lambda)^{(i)}) = \sum_{i=1}^{k} [(\hat{\boldsymbol{\beta}}(\lambda)^{(i)})^T \mathbf{x}_{(i)} - log\{\sum_{j \in \mathcal{R}_{(t_{(i)})}} exp((\hat{\boldsymbol{\beta}}(\lambda)^{(i)})^T \mathbf{x}_j)\}]. \tag{6.5}$$

Estimatet $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ regnes i dette tilfellet ut ved å maksimere den straffede partielle likelihooden for en gitt straffeparameter λ på bakgrunn av et datasett, $data^{(i)}$, der man har endret sensurstatusen til individ i, det vil si at $\delta_{(i)}$ er endret fra 1 til 0. Dette medfører at $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ estimeres på bakgrunn av log-likelihooden der det i'te bidraget er fjernet, men alle de andre bidragene er uendrede. Dette kommer av at individ i bare er sensurert og dermed fremdeles bidrar i risikogruppene til alle individer j som har $t_{(j)} < t_{(i)}$, på samme måte som når individ i var usensurert. Etter estimatet $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ er funnet regner man ut det i'te individets bidrag til log-likelihooden $\varphi_{(i)}(\hat{\boldsymbol{\beta}}(\lambda)^{(i)})$. Dette gjøres for alle de usensurerte individene og summeres opp, slik at man får CV_1 som definert over. Dette gjøres for ulike verdier av λ og man finner den verdien som maksimerer CV_1.

Kuk argumenterer for at dette blir analogt med ordinær kryssvalidering for en generell likelihoodmodell, der effekten av å slette en observasjon er å slette ett bidrag til likelihoodfunksjonen. Ved å kryssvalidere for Cox-modellen på denne måten finner man $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ ved å slette

det i'te individets ledd i log-likelihoodsummen, uten å endre på de andre leddene. På denne måten beholdes informasjonen om at individ i levde fram til $t_{(i)}$, men informasjonen om at vedkommende dør ved $t_{(i)}$ brukes ikke.

6.5.2 Variant 2

Den andre varianten vi skal se på ble foreslått av Verweij og Van Houwelingen i 1993[52]. Deres mål er å finne et kryssvalideringskriterium som kan bestemme hvor god en Cox-modell er til prediksjon av nye observasjoner. De peker på det faktum at komponentene i den partielle likelihooden til denne modellen er avhengige, og at man må ta hensyn til dette dersom man skal gjøre en kryssvalidering. De starter med å presentere uttrykket for den partielle likelihooden:

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \left(\frac{exp(\boldsymbol{\beta}^T \mathbf{x}_i)}{\sum_{j \in R_{(t_i)}} exp(\boldsymbol{\beta}^T \mathbf{x}_j)} \right)^{\delta_i}. \tag{6.6}$$

De ønsker videre å finne det i'te bidraget $L_i(\boldsymbol{\beta})$ til denne partielle likelihooden, og definerer det som $L_i(\boldsymbol{\beta}) = L(\boldsymbol{\beta})/L_{(-i)}(\boldsymbol{\beta})$, der $L_{(-i)}$ er likelihooden beregnet på bakgrunn av datasettet $data^{-i}$ der det i'te individet er fjernet. Dersom tidene t_i er sortert og dersom $\delta_i = 1$ vil $L_i(\boldsymbol{\beta})$ i følge forfatterne være den betingede sannsynligheten for at individ i vil dø ved tid t_i, gitt at individet har overlevd fram til tid t_{i-1}. Dette kommer av at dersom man fjerner den i'te faktoren fra (6.6) vil bidraget fra det i'te individet forsvinne, samtidig som bidragene fra individ i også forsvinner fra alle risikomengder før tid t_i. Dersom alle t_i'ene er sortert, slik at $t_j < t_i$ for $j < i$ blir dette til

$$L_{(-i)}(\boldsymbol{\beta}) = \prod_{j<i} \left(\frac{exp(\boldsymbol{\beta}^T \mathbf{x}_j)}{\sum_{k \in R_{(t_j)}} exp(\boldsymbol{\beta}^T \mathbf{x}_k) - exp(\boldsymbol{\beta}^T \mathbf{x}_i)} \right)^{\delta_j} \prod_{j>i} \left(\frac{exp(\boldsymbol{\beta}^T \mathbf{x}_j)}{\sum_{k \in R_{(t_j)}} exp(\boldsymbol{\beta}^T \mathbf{x}_k)} \right)^{\delta_j},$$

og bidraget $L_i(\boldsymbol{\beta})$ fra individ i til den partielle likelihooden blir som sagt definert som $L(\boldsymbol{\beta})/L_{(-i)}(\boldsymbol{\beta})$. For den tilsvarende log-likelihooden blir dette

$$l_i(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) - l_{(-i)}(\boldsymbol{\beta}), \tag{6.7}$$

der $l_{(-i)}(\boldsymbol{\beta})$ er log-likelihooden beregnet uten det i'te individet i datasettet. Verdien av $\boldsymbol{\beta}$ som maksimerer $l_{(-i)}(\boldsymbol{\beta})$ skrives som $\hat{\boldsymbol{\beta}}^{-i}$. Kryssvalideringskriteriet cvl blir så definert som

$$cvl = \sum_{i=1}^{n} l_i(\hat{\boldsymbol{\beta}}^{-i}).$$

For en gitt modell måler cvl hvor godt observasjon i kan predikeres når man bruker de andre observasjonene til å tilpasse modellen.

Dette forslaget til kryssvalidering kan lett overføres til en straffet log-likelihood for Cox-modellen. Man finner da estimatet $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ for en verdi av straffeparameteren λ ved å maksimere den straffede likelihooden (6.3) og beregner så kryssvalideringskriteriet som

$$CV_2(\lambda|data) = \sum_{i=1}^{n} l_i(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) = \sum_{i=1}^{n} \{l(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - l_{(-i)}(\hat{\boldsymbol{\beta}}(\lambda)^{-i})\} \tag{6.8}$$

Den verdien av λ som maksimerer dette uttrykket vil gi den optimale modellen etter dette kriteriet.

6.5.3 Variant 3

Jeg presenterer et tredje alternativ for å estimere straffeparameteren λ. Motivasjonen for dette alternativet tar utgangspunkt i kryssvalidering for lineære likelihoodmodeller med ridge straff, som vi så på i avsnitt 6.4. Vi så der at dersom leddene i log-likelihoodsummen var uavhengige kunne en kryssvalidering utføres ved å ta ut et og et individ fra datasettet. For hvert utelatte individ estimeres en modell på det resterende datasettet, og så ser man hvor godt det utelatte individet passer til denne estimerte modellen ved å regne ut individets bidrag til den estimerte log-likelihoodmodellen. Disse bidragene kan summeres opp og brukes som kryssvalideringskriterium.

Dette tredje kryssvalideringsalternativet for Cox-modellen går ut på å gjøre akkurat det tilsvarende for Cox-modellen, man ser altså bort fra at de ulike leddene i log-likelihooden kan være avhengige. I dette alternativet estimeres $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ som i variant 2, ved å maksimere den straffede log-likelihooden med $data^{-i}$, det vil si datasettet uten den i'te observasjonen. Deretter bruker man dette i'te individets bidrag til log-likelihooden for å lage kryssvalideringskriteriet:

$$CV_3(\lambda|data) = \sum_{i=1}^{n} \varphi_i(\hat{\boldsymbol{\beta}}(\lambda)^{-i}). \tag{6.9}$$

Dette er det samme kryssvalideringskriteriet som for variant 1, men med et annet estimat for $\boldsymbol{\beta}$. Dette blir analogt med vanlig CV fordi $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ estimeres ved å late som om observasjon i ikke eksisterer, det fjernes fra alle risikogrupper og det i'te bidraget til log-likelihooden faller bort. Etter man har tilpasset modellen finner man så dette i'te bidraget til den estimerte log-likelihooden for å se hvor godt observasjonen passer med den estimerte modellen. I dette kryssvalideringskriteriet ser man altså bort fra mulige bidrag det utelatte individet i kan ha til andre risikogrupper. Ideen er at det i'te log-likelihoodbidraget vil være et tilstrekkelig mål for hvor godt individ passer til den estimerte modellen, og det er ikke nødvendig å korrigere for bidrag i andre ledd.

6.6 Sammenlikning av kryssvalideringsmetoder for Cox-modellen

Vi starter sammenlikningen med å definere *risikoen* til individ i som

$$\omega_i = exp(\boldsymbol{\beta}^T \mathbf{x}_i).$$

Denne definisjonen har sammenheng med at dersom $\boldsymbol{\beta}^T \mathbf{x}_i$ har en stor negativ verdi så har individ i stor sjanse for å oppnå en lang levetid. Dette gjenspeiles i at $exp(\boldsymbol{\beta}^T \mathbf{x}_i)$ blir nær null, det vil si at personen har lav risiko. Hvis $\boldsymbol{\beta}^T \mathbf{x}_i$ øker vil også risikoen ω_i øke. Dersom man studerer Cox-modellen ser man at $\boldsymbol{\beta}$ og $\mathbf{x}_i$ inngår i modellen *kun* på denne formen $\omega_i = \boldsymbol{\beta}^T \mathbf{x}_i$. Dette kan ses på som at det er de ulike individenes risiko man bruker til å estimere modellen. Denne definisjonen av risiko vil bli brukt under sammenlikningen av de ulike metodene for kryssvalidering for denne modellen, blant annet ved å se på hvordan de vektlegger individenes risiko ulikt.

Anta i resten av kapittelet at levetidene er sortert, slik at $t_j < t_i$ for $j < i$. Vi starter med å sammenlikne hvordan de tre ulike metodene beregner det jackknife-liknende estimatet for

$\boldsymbol{\beta}$. Variant 1 finner dette estimatet ved å optimere den straffede log-likelihooden på bakgrunn av datasettet der et av de usensurerte individene har blitt sensurert. Man finner altså den $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ som minimerer

$$l_{(i)}(\boldsymbol{\beta}) = \sum_{j<i} \varphi_j(\boldsymbol{\beta}) + \sum_{j>i} \varphi_j(\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta}. \tag{6.10}$$

Variant 2 og variant 3 estimerer $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ ved å holde individ i ute fra datasettet. Dersom individ i er fjernet kan vi for alle $j < i$ definere det j'te bidraget til log-likelihoodsummen som:

$$\varphi_j^{-i}(\boldsymbol{\beta}) = \delta_j \boldsymbol{\beta}^T \mathbf{x}_j - \delta_j log \left\{ \sum_{k \geq j} exp(\boldsymbol{\beta}^T \mathbf{x}_k) - exp(\boldsymbol{\beta}^T \mathbf{x}_i) \right\}. \tag{6.11}$$

For alle $j > i$ vil $\varphi_j^{-i}(\boldsymbol{\beta})$ være lik $\varphi_j(\boldsymbol{\beta})$ ettersom i'te observasjon bare inngår i risikogruppen til de som har tider mindre enn t_i. Estimatet $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ finnes så ved å maksimere uttrykket

$$l_{(-i)}(\boldsymbol{\beta}) = \sum_{j<i} \varphi_j^{-i}(\boldsymbol{\beta}) + \sum_{j<i} \varphi_j(\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta}. \tag{6.12}$$

Forskjellen mellom disse to uttrykkene for å finne henholdsvis $\hat{\boldsymbol{\beta}}(\lambda)^{(i)}$ og $\hat{\boldsymbol{\beta}}(\lambda)^{-i}$ er at risikoen til det i'te individet har en innvirkning på beregningen av jackknifeestimatet i den første metoden, men ikke i de to andre. Dersom ω_i er liten vil den ha liten innvirkning i variant 1 og de to jackknifeestimatene fra (6.10) og (6.12) vil bli tilnærmet like. Estimatene vil bli forskjellige dersom ω_i er så stor at den gir en betydelig endring i leddene for $j < i$. Personer med høy risiko vil altså kunne påvirke forskjellene mellom de to estimatene mer enn personer med liten risiko.

Dersom man sammenlikner kryssvalideringskriteriene CV_1, CV_2 og CV_3 ser man at variant 1 og variant 3 begge maksimerer summen av de i log-likelihoodbidragene. Forskjellen mellom dem er som tidligere nevnt at de bruker ulike metoder for å estimere $\boldsymbol{\beta}$. Dersom man antar at disse estimatene blir svært like, det vil si at risikoen til det i'te individ har liten innvirkning på estimatet til variant 1, vil man kunne forvente at disse to kryssvalideringsmetodene vil oppføre seg svært likt.

Variant 2 og variant 3 estimerer $\boldsymbol{\beta}$ på samme måte, men har ulike kryssvalideringskriterier. Ved å regne litt på de to kryssvalideringskriteriene vil forskjellen mellom dem komme klarere frem. La $\varphi_j(\boldsymbol{\beta})$ og $\varphi_j^{-i}(\boldsymbol{\beta})$ være definert som tidligere i (6.4) og (6.11). Da kan man skrive ut

uttrykket i for $CV_2(\lambda|data)$ (6.8) som:

$$\begin{aligned} CV_2(\lambda|data) &= \sum_{i=1}^{n} l_i(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) = \sum_{i=1}^{n} \left[l(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - l_{(-i)}(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) \right] \\ &= \sum_{i=1}^{n} \left[\sum_{j=1}^{n} \varphi_j(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - \sum_{j<i} \varphi_j^{-i}(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - \sum_{j>i} \varphi_j(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) \right] \\ &= \sum_{i=1}^{n} \left[\sum_{j\leq i} \varphi_j(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - \sum_{j<i} \varphi_j^{-i}(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) \right] \\ &= \underbrace{\sum_{i=1}^{n} \varphi_i(\hat{\boldsymbol{\beta}}(\lambda)^{-i})}_{(1)} + \underbrace{\sum_{i=1}^{n} \sum_{j<i} \left[\varphi_j(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - \varphi_j^{-i}(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) \right]}_{(2)}. \end{aligned}$$

Dersom man sammenlikner dette uttrykket med kryssvalideringskriteriet CV_3 fra (6.9) ser man at (1) er identisk med CV_3. Forskjellen mellom de to metodene kommer av at variant 3 i tillegg har med (2) i uttrykket over. Dette tillegget kan skrives ut ved å sette inn definisjonene av $\varphi_j(\hat{\boldsymbol{\beta}}(\lambda)^{-i})$ og $\varphi_j^{-i}(\hat{\boldsymbol{\beta}}(\lambda)^{-i})$. Dersom man definerer $\omega_k^{-i} = exp((\hat{\boldsymbol{\beta}}(\lambda)^{-i})^T \mathbf{x}_k)$ får man at dette tillegget blir

$$\begin{aligned} &\sum_{i=1}^{n} \sum_{j<i} \left[\varphi_j(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) - \varphi_j^{-i}(\hat{\boldsymbol{\beta}}(\lambda)^{-i}) \right] = \\ &\sum_{i=1}^{n} \sum_{j<i} \delta_j \left[(\hat{\boldsymbol{\beta}}(\lambda)^{-i})^T \mathbf{x}_j - log \left\{ \sum_{k\geq j} \omega_k^{-i} \right\} - (\hat{\boldsymbol{\beta}}(\lambda)^{-i})^T \mathbf{x}_j + log \left\{ \sum_{k\geq j} (\omega_k^{-i} - \omega_i^{-i}) \right\} \right] = \\ &\sum_{i=1}^{n} \sum_{j<i} \delta_j log \left\{ \frac{\sum_{k\geq j} (\omega_k^{-i} - \omega_i^{-i})}{\sum_{k\geq j} \omega_k^{-i}} \right\} \end{aligned}$$

. Definer

$$D_j = \delta_j \; log\{ \underbrace{\frac{\sum_{k\geq j} (\omega_k^{-i} - \omega_i^{-i})}{\sum_{k\geq j} \omega_k^{-i}}}_{(d_j)} \}.$$

Tillegget for individ i blir da $\sum_{j<i} D_j$. Uttrykket (d_j) vil ta verdier mellom 0 og 1, avhengig av hvor stor den estimerte risikoen ω_i^{-i} er. Dersom risikoen ω_i^{-i} er svært liten vil (d_j) bli nær 1 og alle D_j bli svært små. Jo større ω_i^{-i} er, jo mindre vil (d_j) bli og D_j vil få en negativ verdi.

Kort oppsummert har vi sett at CV-variant 2 og 3 estimerer $\boldsymbol{\beta}$ likt, men at i CV_2 vektlegges risikoen til individ i som holdes utenfor i langt større grad enn i CV_3. På bakgrunn av disse ulikhetene vil det være rimelig å kunne forvente at disse to variantene vil gi ulike resultater. Variant 1 og 3 har det samme kryssvalideringskriteriet, men bruker ulike metoder for å estimere $\boldsymbol{\beta}$. Som vi så er disse to estimeringsmetodene for $\boldsymbol{\beta}$ rimelig like, så det ville være nærliggende å tro at disse to variantene vil kunne gi liknende resultater. Det er vanskelig å si ut fra disse teoretiske betraktningene hvilke av disse kryssvalideringsvariantene som er mest hensikstmessig å bruke. Det vil derimot være mulig å teste dem ut på ulike datasett, og sammenlikne dem etter ulike kriterier. Resultater av dette vil bli presentert i kapittel 8.

6.6.1 Ridge trace

For å kunne lage et kryssvalideringsplott slik at man kan plukke ut $\hat{\lambda}$ er det nødvendig å plotte kryssvalideringskriteriet for verdier av λ i et intervall der man kan være rimelig sikker på at den optimale $\hat{\lambda}$ ligger. Kryssvalidering er ressurskrevende, ettersom det krever mange modellestimeringer, så det vil være fornuftig med tanke på beregningstid å velge et intervall som både inneholder $\hat{\lambda}$ og som i tillegg har god nok oppløsning til at verdien til denne optimale parameteren kan bestemmes med tilstrekkelig nøyaktighet.

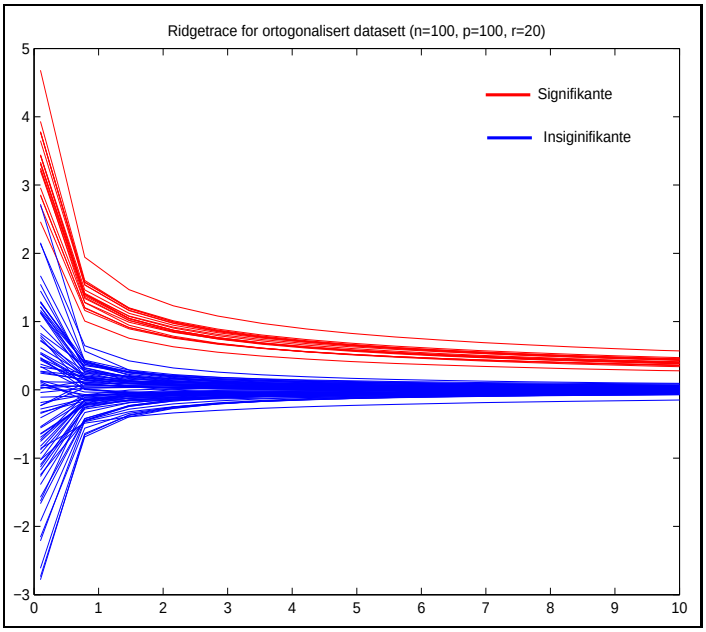
Et hjelpemiddel for å velge hvilket intervall man bør lete etter $\hat{\lambda}$ i er et såkalt $\boldsymbol{\beta}$*-trace* eller *ridgetrace*. Dette er et todimensjonalt plott av komponentene $\beta_{(i)\lambda}$ for $i = 1, \ldots, p$ mot u ulike verdier av λ i intervallet $[a, b]$. Ved å inspisere tracet kan man få en ide om hvor sensitivt systemet er, og dermed for hvilke verdier av λ man kan finne rimelig stabile estimater for $\boldsymbol{\beta}$. Ved å lage et ridgetrace for et rimelig vidt intervall $[a, b]$ kan man bruke dette til å velge seg et mindre intervall $[c, d]$ der man vil kunne anta at verdien $\hat{\lambda}$ ligger. Dette mindre intervallet kan så brukes til å gjøre en kryssvalidering. Ved å se på ridgetracet kan man også se hvilke komponenter av $\hat{\boldsymbol{\beta}}$ som krympes mot 0 først ettersom λ økes. Tracet kan altså gi hint om hvilke koeffisienter som er signifikante, men dette blir mer problematisk å se ettersom antall koeffisienter øker.

Ridgetrace ble foreslått av Hoerl og Kennard[22] som en måte å velge ut den optimale straffeparameteren i multikollineære systemer. Ved å inspisere et slikt trace kunne man finne $\hat{\lambda}$ som den parameteren der koeffisientverdiene stabiliserte seg på rimelige verdier.

Figur 6.3 viser et eksempel på et ridgetrace for en cox-modell på simulerte data. Der ser man at de estimerte koeffisientene har for store verdier når straffen er 0, men ettersom straffeparameteren økes krympes verdien til koeffisientene. De røde strekene tilhører signifikante koeffisienter, som skal ha verdi 1, og de blå strekene tilhører insignifikante koeffisienter som skal ha verdi 0. De estimerte koeffisientverdiene stabiliserer seg på rimelige verdier for λ mellom 1 og 4. Det vil være naturlig å kjøre en kryssvalidering for disse dataene med dette intervallet.

6.7 L-kurven

Optimal straffeparameter $\hat{\lambda}$ for en generell regulariseringsmetode kan også finnes ved hjelp av det såkalte *L-kurve kriteriet*. En L-kurve for et ordinært lineært regresjonsproblem er et plott av normen til den regulariserte parametervektoren $||\boldsymbol{\beta}_\lambda||_2$ mot den korresponderende residualnormen $||X\boldsymbol{\beta} - \mathbf{y}||_2$. L-kurven viser kompromisset mellom å minimere disse to kvantitetene, noe som er essensen i de fleste regulariseringsmetoder. L-kurven er kontinuerlig dersom λ er kontinuerlig, slik tilfellet er i for eksempel ridge regresjon. For diskrete ill-posed problemer viser det seg at L-kurven, når den plottes i en log-log skala, ofte følger en karakteristisk L-formet kurve, med et klart hjørne som skiller de vertikale og de horisontale delene av kurven. Bruk av log-log gjør at denne karakteristiske L-formen kommer klarere fram. Den vertikale delen av kurven representerer området der $||\hat{\boldsymbol{\beta}}||_2$ er sensitiv for endringer i λ. Her er λ for liten og løsningen vil domineres av perturbasjonsfeil. En slik situasjon kalles *underglatting*. Den horisontale delen av kurven representerer området der $||X\boldsymbol{\beta} - y||_2$ er sensitiv for endringer i λ. Her blir λ for stor og løsningen domineres av regulariseringsfeil. Denne situasjonen

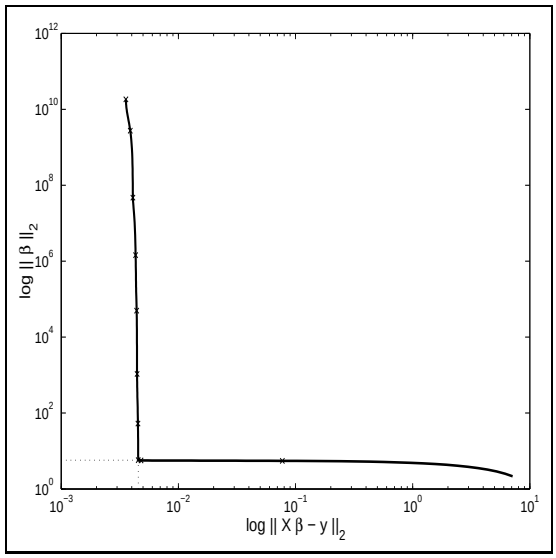


Figur 6.3: Eksempel på et ridgetrace for den straffede Cox-modellen med 100 observasjoner, 100 kovariater der 20 av kovariatene har betydning. Kurvene stabiliseres når $\lambda > 4$, så det kan være fornuftig å lete etter optimal straffeparameter i intervallet $[0, 4]$.

kalles *overglatting*. Et spesielt trekk ved L-kurven er at den optimale reguleringsparameteren ligger nær den reguleringsparameteren som korresponderer med L-kurvens hjørne. Det er denne viktige egenskapen ved L-kurven som er grunnlag for valg av reguleringsparameter. Velger man λ fra hjørnet i kurven kan man finne en regularisert løsning med god balanse mellom perturbasjonsfeil og reguleringsfeil. L-kurvens hjørne defineres som det punktet der kurven $(log||X\boldsymbol{\beta} - y||, log||\boldsymbol{\beta}||)$ har maksimal krumning. Man kan altså finne den optimale straffeparameteren $\hat{\lambda}$ ved å finne den λ som maksimerer denne krumningen. Se figur 6.4 for et eksempel på en L-kurve for et enkelt lineært problem. Kurven er laget for et test-problem innen bildeanalyse og har en spesielt klar L-form.

En L-kurve kan også overføres til en generell modell, som tilpasses ved hjelp av likelihoodestimering. Da plotter man som før logaritmen av normen til den straffede regresjonsparameteren, $||\hat{\boldsymbol{\beta}}_(\lambda)||$, men i dette tilfellet plottes den mot logaritmen av den negative log-likelihooden beregnet for de ulike verdiene av λ, ettersom det er denne man ønsker å minimere når man har en likelihoodbasert estimering. Optimal straffeparameter $\hat{\lambda}$ kan da finnes på samme måte som i det lineære tilfellet, ved å finne den λ som maksimerer krumningen på kurven.

L-kurvekriteriet er ikke en standard modellseleksjonsmetode for statistiske modeller. Den har sitt opphav i en annen kultur, nemlig studier av regularisering og studier av ill-posed inverse problemer. Den er såvidt meg bekjent ikke omtalt tidligere i statistisk sammenheng, men det vil likevel kunne være interessant å teste det ut og evaluere ytelsen til denne metoden i en regularisert statistisk modell som den vi ser på i denne oppgaven.



Figur 6.4: Eksempel på en L-kurve for et enkelt lineært problem. Kurvens maksimale krumningspunkt er merket med stiplede linjer.

Kapittel 7

Genseleksjon

7.1 En enkel tilnærming

Det er naturlig å spørre om det er mulig å gjøre en genseleksjon på bakgrunn av den statistiske analysen av overlevelsesdataene som vi har sett på i tidligere kapitler. Målet vil da være å plukke ut de genene som har sterk assosiasjon med overlevelse. Videre studier av disse genene vil kunne gi informasjon om funksjonen til disse genene og hvorfor akkurat disse genene er relevante for overlevelsen.

Betrakt Cox-modellen i likning (3.8) og anta at det er p kovariater. En naiv tilnærming til å gjøre en variabelseleksjon vil være å studere den estimerte koeffisientvektoren $\hat{\boldsymbol{\beta}}$ fra Cox-modellen. Hver estimerte koeffisient $\hat{\beta}_i$ antas å ha informasjon om relevansen for det i'te gen på overlevelsen. Løselig har man at dersom $\hat{\beta}_i \approx 0$ vil dette indikere at gen i har liten eller ingen innvirkning på overlevelsen, og når $|\hat{\beta}_i|$ er stor indikerer dette at gen i har stor betydning for prediksjon av levetiden. Dersom man sier at størrelsen på absoluttverdien til $\hat{\beta}_i$ bestemmer hvor stor innvirkning genet har på overlevelsen kan en variabelseleksjon gjøres enkelt ved først å velge oss en terskelverdi, t, og så plukke ut de r genene som har koeffisientverdier $\beta_i > t$, som de genene som har relevans for prediksjon av overlevelsen. Terskelverdien kan for eksempel velges som en fast verdi bestemt ut fra å studere størrelsene på komponentene i $\hat{\boldsymbol{\beta}}$, slik at man får med et visst antall gener, r.

En klar ulempe med denne tilnærmingen er at det ikke tas hensyn til at de ulike estimatene, $\hat{\beta}_i,\ i = 1, \ldots, p$, har ulik varians. Stor varians til et estimert koeffisient vil kreve at koeffisienten har en stor absoluttverdi før den vil kunne kalles interessant. Snarere enn å velge en felles terskel for alle $\hat{\beta}_i$ bør en derfor finne separate terskler $t_i,\ i = 1, \ldots, p$ for hver $\hat{\beta}_i$ som tar hensyn til variansen i hvert estimat. En måte å gjøre dette på er ved å utføre hypotesetesting.

7.2 Hypotesetesting

For en estimert Cox-modell ønsker man som sagt å kunne bestemme hvilke av kovariatene som har betydning for responsen, det vil si som har betydning for levetiden til et individ. For

	# positive utfall	# negative utfall	# totalt
Ekte negative	F	p_0 - F	p_0
Ekte positive	T	p_1 - T	p_1
Totalt	S	p - S	p

Tabell 7.1: Oppsummering av mulige utfall av hypostesetester.

hvert gen ønsker man å teste hypotesen

$$H_0 : \textit{Genet har ingen betydning},$$

mot den alternative hypotesen

$$H_a : \textit{Genet har betydning}.$$

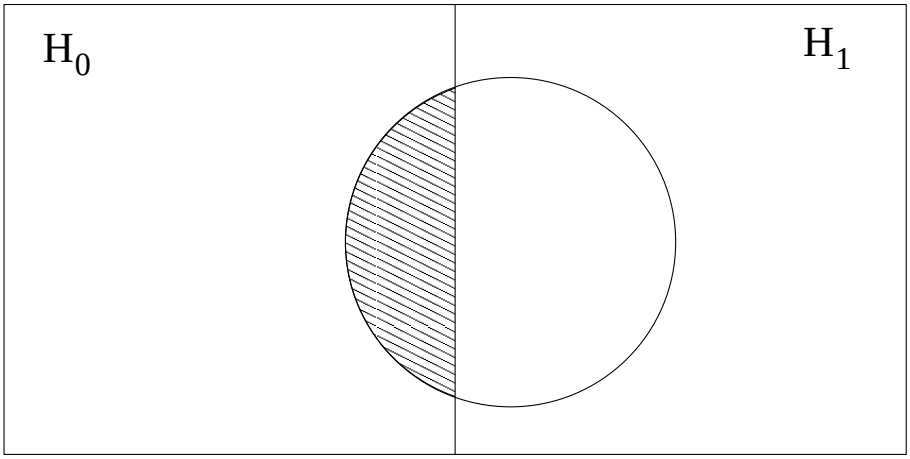
Målet med å teste en hypotese er å gjøre et valg. Man sier at testen gir *positivt* resultat dersom H_0 forkastes til fordel for den alternative hypotesen. Dersom H_0 virkelig er sann for et slikt gen kalles dette en *falsk positiv* (type 1 feil), ellers kalles det en *ekte positiv*. Dersom vi derimot velger å beholde H_0 for dette genet kalles det en *negativ*, og med tilsvarende notasjon kalles det en *ekte negativ* dersom H_0 virkelig er sann for dette genet, og en *falsk negativ* (type 2 feil) ellers. I praksis vet man ikke om H_0 er sann, så det er ikke mulig å si om man har gjort riktig valg. Det er likevel mulig å lage gode tester som medfører at sjansen for å gjøre slike feil blir liten. Tabell 7.1 viser oversikt over mulige utfall av p hypotesetester. Kort oppsummert er F antall falske positive utfall, T er antall ekte positive utfall mens S er totalt antall positive utfall.

En mulighet for å teste hvilke gener som har relevans for overlevelsen innebærer å sette opp nullhypoteser for de ulike genene for å teste om koeffisienten til hvert enkelt gen er ulik 0. Man må da utføre en rekke hypotesetester på formen:

$$\begin{aligned} H_0 :&\ \beta_1 = 0 \\ H_0 :&\ \beta_2 = 0 \\ &\vdots \\ H_0 :&\ \beta_p = 0. \end{aligned}$$

Hver hypotese testes ved å regne ut en testobservator T for den estimerte koeffisienten $\hat{\beta}_i$, for eksempel ved å sammenlikne selve estimatet med variansen til estimatet. Sannsynlighetsfordelingen til T under H_0 kalles *nullfordelingen* og basert på dennne er det mulig å beregne en såkalt *p-verdi* for testobservatoren. Denne p-verdien sier hvor sannsynlig det er at man ville observert denne eller en større verdi for β_i dersom H_0 var sann. Jo mindre p-verdien er, jo sterkere er indisiene mot H_0. En vanlig tilnærming til hypotesetester er å si at H_0 forkastes dersom p-verdien er mindre enn en gitt verdi α, som også kalles for signifikansnivået for testen.

I mikroarrayeksperimenter er antall gener stort og man ønsker derfor å teste mange hypoteser, og en god del av disse kan være ekte positive. Ønskesituasjonen ville være om man klarte å plukke ut alle de ekte positive, det vil si alle genene som har betydning for overlevelsen, og ingen falske positive. Det er i praksis umulig å forsikre seg om at dette er tilfellet. Et mer



Figur 7.1: Hele firkanten illustrerer universet av hendelser, delt inn i signifikante H_1 og insignifikante H_0. Sirkelen representerer mengden av hendelser som kommer ut som signifikante under hypotesetesting. Det skraverte feltet er de hendelsene som kommer ut som signifikante, men som egentlig ligger i H_0. Disse er såkalte *falske positive*. Resten av sirkelen er ekte positive, det vil si de hendelsene vi er interessert i å finne.

realistisk mål kan isteden være å finne så mange av de ekte positive som mulig, samtidig som man aksepterer noen få falske positive. Man ønsker likevel at andelen falske positive skal være liten i forhold til totalt antall gener som kommer ut som positive. Dette skyldes blant annet at man ved å plukke ut et stort antall falske positive vil kunne kaste bort mye unødvendig tid og ressurser på videre studier av gener som ikke har noen relasjon til responsen man er interessert i å studere.

Et problem som oppstår når man ser på mange tester samtidig er at antallet falske positive ofte blir stort. For et valgt signifikansnivå, si $\alpha = 0.05$ ønsker man at sannsynligheten for at man plukker ut en falsk positiv skal være 0.05. Når man utfører p uavhengige tester har man at $P(\text{minst en falsk positiv}) = 1 - (1 - \alpha)^p$, og denne blir tilnærmet lik 1 når p blir stor. Når man gjør mange tester vil sjansen for å gjøre feil øke med antall tester. Hvis hver test har signifikansnivå $\alpha = 0.05$, det vil si $P(p_i < 0.05) = 0.05$ for $i = 1, 2, \ldots, p$, så vil man for p slike tester ha at

$$E[\#\text{falske positive}] \leq 0.05 * p,$$

altså en større sannsynlighet for å plukke ut flere falske positive jo flere tester som utføres. Anta for eksempel at vi har 10000 gener og at ingen av dem er ekte positive. Ved å teste hvert gen for seg vil vi for hver test ha en 5% sjanse for å gjøre feil. For 10000 individuelle tester vil man forvente at $10000 * 0.05 = 500$ gener har en p-verdi < 0.05, et forholdsvis høyt tall. Figur 7.1 illustrerer dette problemet. Dersom man velger å bruke individuelle p-verdier for hvert gen kontrollerer man raten for at gener som egentlig tilhører H_0 kommer ut som positive. Dette tilsvarer å justere størrelsen på det skraverte feltet i figuren i forhold til størrelsen på H_0. Man ønsker bare at en viss andel, α, av genene i H_0 skal komme ut som positive i testene. Dersom antall gener i H_0 er stort, det vil si at man har mange gener uten betydning for levetiden, vil dette kunne medføre at man får et stort antall falske positive, og det vil være umulig å plukke ut eventuelle gener som egentlig har en effekt blant disse.

En mulig løsning på dette problemet er å justere signifikansnivået etter hvor mange parallelle hypotesetester som utføres. Vi har sett at det å bruke en enkel p-verdi terskel for p tester kun garanterer at $E[\#\text{false positive}] \leq \alpha p$. Dette tallet blir alt for stort når p er stor, og kravet er for liberalt. En måte å kontrollere den såkalte *family-wise error rate (FWER)*, definert som $P(\#\text{false positive}) \geq 1$, er ved å gjøre en såkalt *Bonferroni-korreksjon*. Dette innebærer at man senker signifikansnivået til $\alpha_p = \alpha/p$. Kun tester som gir p-verder under α_p vil forkaste nullhypotesen. Man kan altså garantere at $P(\#\text{false positive} \leq 1) \leq \alpha$ ved å si at de positive genene er de som har $p_i \leq \alpha/p$. En slik korreksjon gir det ønskede signifikansnivået α, men er meget konservativ. Når antall tester blir stort vil det justerte signifikansnivået kunne bli så lavt at ingen tester vil klare å passere, selv om man virkelig har gener som er ekte positive. For genomstudier vil en slik korreksjon som regel bli alt for konservativ, ettersom man forventer at en del av genene har en betydning og man ønsker å finne alle disse ekte positive genene. Det finnes en rekke andre tilnærminger til å kontrollere FWER ved å justere av p-verdiene som er mindre konservative enn dette, men de fleste lider av at sensitiviteten blir for lav ettersom de blir mer spesifikke. Spesielt i mikroarrayeksperimenter der antall tester er stort og antall observasjoner er få har ofte slike justerte tester en tendens til å plukke ut veldig få, om noen gener. Et mindre konservativt alternativ til å gjøre slike justerte tester er å bruke et kriterium som kalles *positive false discovery rate (pFDR)*, som ofte gir gode resultater i situasjoner der man har mange tester.

7.3 False discovery rate

Vi har sett at i situasjoner der antall tester er stort som for eksempel i mikroarrayeksperimenter, er FWER-kontrollerende justeringer av p-verdiene for konservative, mens det å kjøre p paralelle tester vil kunne lede til for mange falske positive. Som en mellomting mellom disse tilnærmingene foreslo Benjamini og Hochberg[6] å heller kontrollere *false discovery rate (FDR)*. FDR er definert som den forventede andelen av falske positive blant alle positive funn, det vil si:

$$FDR = E[\frac{F}{S}|S > 0] \cdot P(S > 0). \tag{7.1}$$

FDR kontrollerer antall falske positive i den forstand at jo flere hypoteser som er ekte falske, jo mindre blir FDR.

Storey og Tibshirani[48] foreslo en modifisert variant av FDR, kalt *positive false discovery rate (pFDR)*. Kriteriet pFDR vektlegger det faktum at en justering av p-verdiene er nødvendig bare så lenge det faktisk er positive funn. Ideen er å bruke feilmålet

$$pFDR = E[\frac{F}{S}|S > 0],$$

der F er antall falske positive og S er antall positive totalt. pFDR er altså forventet andel falske positive blant alle positive. Det at $pFDR \leq \alpha$ for et gitt nivå $0 < \alpha < 1$ vil altså si at av alle genene som kalles positive vil omlag $\alpha\%$ forventes å være falske positive. Dette kan illustreres med den samme figur 7.1 som ble brukt for å illustrere p-verdiene. Det som gjøres ved å bruke pFDR er at størrelsen på det skraverte feltet kontrolleres i forhold til størrelsen på hele sirkelen (det vil si antall falske positive i forhold til antall ekte positive). Bare en andel α av sirkelen får ligge i H_0, og pFDR vil derfor effektivt kontrollere antall falske positive uavhengig av størrelsen på H_0.

Man kan estimere pFDR for en gitt terskel $0 < t \leq 1$, der man kaller alle gener med p-verdier mindre enn t for signifikante. Når p blir stor kan det argumenteres for at følgende approksimasjon holder [48]:

$$pFDR(t) = E\left[\frac{F(t)}{S(t)}\right] \approx \frac{E[F(t)]}{E[S(t)]}.$$

Et enkelt estimat for $E[S(t)]$ er antall observerte p-verdier som er mindre eller lik t. Forventningen til $F(t)$ kan estimeres som $E[F(t)] = p_{(0)}t$, ettersom $P(p_i \leq t) = t$, der $p_{(0)}$ er totalt antall gener som har innvirkning på levetiden. Denne størrelsen er ukjent og må estimeres. Et konservativt estimat er å sette $p_{(0)} = p$, det vil si at man antar at en neglisjerbar andel av genene har noen innvirkning på levetiden. (Et forslag til et mindre konservativt estimat for $p_{(0)}$ kan finnes i [48].) En estimator for $pFDR$ for en terskel t kan da skrives som

$$\widehat{pFDR}(t) = \frac{pt}{\#\{p_i \leq t\}}. \tag{7.2}$$

En mulighet for praktisk bruk av pFDR er å velge en akseptabel feilrate α før man tester, og så estimere p-verditerskelen for forkastning av nullhypotesen. Dette kan lede til at man i noen tilfeller identifiserer bare noen få gener. En bedre framgangsmåte er å estimere pFDR for alle mulige terskler t mellom 0 og 1 samtidig, og så velge den terskelen som best balanserer antallet identifiserte gener og den kontrollerte feilraten. Med andre ord finner man den t^* som minimerer (7.2) og sier at alle gener med p-verdier som er mindre enn t^* er positive.

7.4 Genseleksjon for en straffet Cox-modell

De tilnærmingene til genseleksjon som vi har sett på til nå, nemlig hypotesetesting og pFDR, forutsetter korrekt beregnede p-verdier for estimatene $\hat{\beta}_i$, $i = 1, \ldots, p$. For en tradisjonell Cox-modell vil det være mulig å regne ut slike p-verdier ved å gjøre såkalte lokale tester, der man for en subvektor $\boldsymbol{\beta}_1$ av $\boldsymbol{\beta}$ kan teste nullhypotesen $H_0 : \boldsymbol{\beta}_1 = \mathbf{0}$ mot den alternative hypotesen $H_a : \boldsymbol{\beta}_1 \neq \mathbf{0}$. Dette kan for eksempel gjøres ved å bruke lokale varianter av Wald testen, likelihood-ratio testen eller score testen [28] for hvert gen i, og på denne måten finne en p-verdi for hvert gen.

I og med at det i denne oppgaven benyttes en straffet variant av Cox-modellen vil det derimot ikke være mulig å bruke slike standard tester direkte. Regulariseringen endrer estimatene noe og derfor vil også variansestimeringen for de straffede estimatene bli anderledes enn i det klassiske tilfellet. Dette problemet kan angripes på flere måter. En naiv tilnærming til problemet vil være å anta at regulariseringen har så liten effekt på de straffede estimatene at de kan behandles på samme måte som de ustraffede estimatene. Da kan man benytte de standardtestene som finnes for Cox-modellen, uten å måtte ta hensyn til at man jobber med en regularisert modell. Dersom antagelsen om at regulariseringen har liten innvirkning på estimatene ikke stemmer er dette derimot en dårlig løsning på problemet. Før man kan si noe sikkert om dette må man se nærmere på egenskaper ved slike regulariserte estimater for en straffet log-likelihood. Det er meg bekjent lite litteratur som omhandler dette temaet. En årsak til dette kan være at regularisering ofte har vært benyttet som et verktøy for å forbedre

prediksjon for en modell, og at man derfor har vært mindre interessert i å gjøre inferens rundt de enkelte parameterestimatene.

Det er ikke klart hvordan slik variansestimering bør gjøres og vi vil derfor ikke se nærmere på beregning av p-verdier ved hjelp av klassiske tester for en straffet Cox-modell i denne oppgaven. Et forslag til videre arbeid rundt dette vil bli gitt i diskusjonen. Vi skal isteden se på to alternative tilnærminger til å bestemme om et gen har betydning for levetiden eller ikke.

7.4.1 Permutasjonstest

Det første forslaget går ut på å bruke en form for permutasjonstest. Denne tar utgangspunkt i at man har funnet estimatet $\hat{\boldsymbol{\beta}}^{obs}$ for de observerte dataene og ut fra denne ønsker å si om det i'te gen er relevant for overlevelsen ved å se på $\hat{\beta}_i^{obs}$. Ettersom man i denne situasjonen ikke kjenner fordelingen til dette estimatet under nullhypotesen, er det vanskelig å si om dette er en ekstrem observasjon i forhold til denne fordelingen eller ikke. Ideen bak en permutasjonstest er at dersom nullhypotesen er sann vil alle mulige permutasjoner av data være like sannsynlige. Den observerte verdien av genekspresjonsverdiene for gen i, $\mathbf{x}_i$, kunne tatt en hvilken som helst verdi, uten at dette ville hatt noen innvirkning på overlevelsen. Ved å regne ut $\hat{\beta}_i^{perm}$ for alle mulige permutasjoner av $\mathbf{x}_i$ ville man derfor kunne se om $\hat{\beta}_i^{obs}$ er en typisk verdi i fordelingen til β_i under nullhypotesen. Dersom dette ser ut til å ikke stemme kan det ansees som indisier mot nullhypotesen.

Ettersom man her ser på p ulike gener og ønsker å teste sammenhengen mellom responsen og ett og ett gen for seg, vil dette bli gjort ved å permutere hver kovariat for seg mens resten av kovariatene holdes fast. På denne måten brytes forbindelsen mellom den i'te kovariaten og responsen, mens forholdet mellom resten av kovariatene og responsen forblir uendret. Denne måten endrer likevel noe på kovariatstrukturen i dataene. Eventuelle korrelasjoner mellom for eksempel to av kovariatene kan forsvinne når en av disse permuteres. Det vil være rimelig å tro at slike små endringer i kovariatstrukturen ikke vil spille en stor rolle i en slik test som er beskrevet, men bør testes ut før man kan si noe sikkert om dette er tilfellet. Ettersom man ønsker å teste hvert gen for seg er det viktigst at man ikke ødelegger sammenhengen mellom responsen og de andre kovariatene, ettersom disse også forklarer en del av responsen.

En prosedyre for å teste om gen i er relevant for overlevelsen ved å gjøre permuteringen vil omfatte følgende steg:

1. Permuter komponentene i $\mathbf{x}_i$ og la alle andre $\mathbf{x}_j,\ i \neq j$, og responsen, være uendret.
2. Beregn $\hat{\boldsymbol{\beta}}^{perm}$ for det permuterte datasettet og plukk ut $\hat{\beta}_i^{perm}$ fra denne.
3. Gjenta steg 1 og 2 tilstrekkelig mange ganger.
4. Tell antall ganger $\hat{\beta}_i^{perm}$ er større enn $\hat{\beta}_i^{obs}$ og del dette antallet på antall permutasjoner som ble gjort. Dette gir andelen a^{perm} som sier hvor stor andel av permutasjonene som ga en verdi $\hat{\beta}_i^{perm}$ større enn $\hat{\beta}_i^{obs}$.

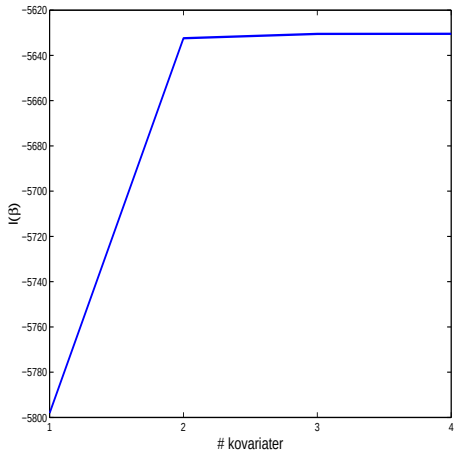
Andelen a^{perm} er da et mål på hvor sannsynlig det er å observere $\hat{\beta}_i^{obs}$ under nullhypotesen. Dersom det ikke er praktisk mulig å gjøre dette for alle mulige permutasjoner av $\mathbf{x}_i$ er det mulig å gjøre det samme ved å bruke et stort nok antall tilfeldige permutasjoner.

7.4.2 Log-likelihoodplott

Som et siste forslag til genseleksjon vil en rimelig enkel metode bli presentert. Denne tar utgangpunkt i det straffede estimatet $\hat{\boldsymbol{\beta}}(\lambda)$ for den fulle straffede modellen. Man starter med å plukke ut kovariaten som tilsvarer den $\hat{\beta}_i(\lambda)$ med størst absoluttverdi og estimerer en modell på bakgrunn av denne ene kovariaten. Deretter plukker man ut kovariaten som tilsvarer den estimerte koeffisienten med nest størst absoluttverdi, og tilpasser en ny modell på bakgrunn av begge de to utvalgte kovariatene. Begge parameterne estimeres altså på nytt i den nye modellen. Slik fortsetter man med å inkludere en og en kovariat i minkende rekkefølge basert på absoluttverdien til koeffisientene, og tilpasser en ny modell for hver kovariat som inkluderes. Dette resulterer i p estimerte modeller, og for hver modell regner man ut log-likelihoodverdien.

Dersom de r første genene som inkludres i modellen er de med størst innvirkning på levetiden vil man kunne forvente at log-likelihoodverdien til de utvidede modellene vil øke inntil disse r genene er inkludert og så stabilisere seg (eller øke mindre) ettersom resten av genene inkluderes. Ved å inspisere et plott over log-likelihoodverdiene for de p modellene vil man derfor kunne forvente å se et slags knekkpunkt når de r genene med størst betydning for levetiden har blitt inkludert i modellen.

Som et enkelt eksempel kan vi se på en modell tilpasset på bakgrunn av 4 gener. Estimering av den fulle straffede modellen gir et straffet parameterestimat $\hat{\boldsymbol{\beta}}(\lambda) = (3.06, 3.22, -0.04, -0.05)$. Ved å bruke metoden over tilpasser man da fire modeller, den første med bare gen 2, så en ny modell med gen 2 og gen 1, en tredje modell med gen 2, gen 1 og gen 4 og til slutt en modell med alle genene. For hver modell regnes log-likelihoodverdien ut, og denne plottes mot antall gener i modellen. Figur 7.2 viser log-likelihoodplottet for dette eksempelet. Vi ser at effekten av å inkludere de to første genene gjør at likelihoodverdien øker, mens inkludering av de to siste genene har liten effekt på denne. På bakgrunn av dette kan vi velge ut gen 2 og gen 1 som de med innvirkning på levetiden.



Figur 7.2: Figuren viser plott over log-likelihoodverdier for eksempelet.

Kapittel 8

Resultater

Den straffede Cox-modellen som har blitt presentert i de foregående kapitlene ble tilpasset på bakgrunn av ulike overlevelses- og genekspresjonsdata. Resultatene vil bli presentert i dette kapittelet. Det består av fire hovedseksjoner. Den første gir en oversikt over de ulike typene data som ble brukt for å teste metodene. Den andre delen går ut på å velge estimeringsmetode for Cox-modellen. I den tredje delen ser vi på hvordan en straffet Cox-modell kan tilpasses til ulike datasett når straffeparameteren λ holdes fast. Den siste delen vil ta for seg testing av ulike metoder for modellseleksjon, det vil si at man ønsker å estimere straffeparameteren λ som gir en best mulig modell.

8.1 Presentasjon av data

Resultatene i dette kapittelet er basert på testing med ulike typer data. Vi skal i dette avsnittet se nærmere på de ulike simulerte og ekte data som ble benyttet.

8.1.1 Simulerte data

Bruk av simulerte (kunstig genererte) data gjør det mulig å ha kontroll over ulike variable, som for eksempel antall gener som brukes til å simulere overlevelsestider, antall individer som er med, hvor mye støy det skal være i dataene, hvor mye korrelasjon det skal være mellom genene og annet. Fordelen med å simulere er at man kjenner den sanne modellen, og dermed får en ide om hvor godt metodene fungerer i ulike situasjoner.

Datasettene ble simulert fra en Weibullfordeling ved den metoden som ble presentert i avsnitt 3.8. Denne tar blant annet utgangspunkt i et datasett med kovariater som skal representere genekspresjonsdataene. Det ble benyttet to metoder for å lage slike kovariatdatasett:

Metode 1:
Den første metoden gikk ut på å lage et datasett der genene var korrelerte. Et datasett $X \in \mathbb{R}^{n \times p}$ ble trukket slik at alle ekspresjonsverdiene x_{ij} er uavhengig identisk fordelte,

$x_{ij} \sim N(0,1)$. Den i'te raden i X representerer genekspresjonsvektoren til individ i og j'te kolonne i X representerer ekspresjonen av gen j for alle n individer. Ettersom alle de p genekspresjonsvektorene $\mathbf{x}_i$ har endelig lengde n fører dette nødvendigvis til at noen av kolonnene (genene) i datamatrisen X vil kunne bli mer eller mindre sporadisk korrelerte. Korrelasjon mellom genekspresjonsvektorene er et trekk man finner igjen hos reelle genekspresjonsdata.

Metode 2:
Den andre metoden gikk ut på først å trekke et datasett X fra standardnormalfordelingen på samme måte som i metoden over, det vil si slik at x_{ij} er uavhengig identisk fordelte fra $N(0,1)$. Deretter ble korrelasjonen mellom de r første kolonnene (genene) i datasettet fjernet ved å ortogonalisere dem på hverandre, ved hjelp av en Gram-Schmidt prosedyre (som beskrevet i blant annet [30]). De $(p-r)$ siste kolonnene ble også gjort ortogonale på de r første kolonnene, men de ble ikke gjort innbyrdes ortogonale. Ved å ortogonalisere kovariatene på denne måten vil man skape minimalt med støy i retningene til de r første genene, ettersom ekspresjonsvektorene til disse r genene vil være lineært uavhengige av ekspresjonsvektorene til alle de andre genene. Dette gir en tilnærmet ideell situasjon der effekten av de r første genene ikke forstyrres av effekten av resten av genene.

På bakgrunn av slike datasett med kovariater ble det så generert levetider. Disse ble trukket fra en Weibullfordeling, slik det ble beskrevet i avsnitt 3.8. Grunnen til at Weibullfordelingen ble valgt er at det er en vanlig fordeling å anta for overlevelsestider, og fordi den gir en enkel måte å innføre kovariater i en proporsjonal hasardmodell. I tillegg er fordelingen veldig fleksibel. Blant annet kan hasardraten for denne modellen anta flere ulike former, både konstant, økende og minkende. Denne modellen vil være tilstrekkelig fleksibel for å kunne simulere den typen data som trengs i denne oppgaven.

Levetidene ble simulert slik at de r første genene, det vil si de r første kolonnene i X, skulle ha betydning for levetidene. Hvor stor betydning de hadde ble bestemt av parameteren s. Legg merke til at det ikke er mulig å gi noen av genene mer betydning enn andre ved bruk av denne metoden. De $p-r$ siste genene hadde ingen innvirkning på de simulerte levetidene. Dette tilsvarer at den sanne koeffisientvektoren $\boldsymbol{\beta}$ som ble brukt til å simulere levetidene har komponenter $\beta_1 = \ldots = \beta_r = s$ og $\beta_{r+1} = \ldots = \beta_p = 0$. Det ble også gjort mulig å velge hvor stor andel av individene som skulle være usensurerte. Parameteren $0 \leq z \leq 1$ bestemte dette ved at dersom $z = 1$ så ble ingen individer sensurerte, mens dersom $z = 0$ så ble alle individene sensurerte.

Resultatet av simuleringene var datasett som videre vil betegnes som *ikke-ortogonalisert* dersom kovariatene ble generert med metode 1 over, og *ortogonalisert* dersom kovariatverdiene ble generert med metode 2. Datasettene består av en kovariatmatrise $X \in \mathbb{R}^{n \times p}$ og en responsmatrise $Y \in (\mathbb{R} \times \{0,1\})^n$. Denne responsmatrisen består av de n simulerte levetidene med tilhørende sensurstatus, det vil si $Y = ((t_1, \delta_1), \ldots, (t_n, \delta_n))^T$. Etter at et slikt datasett er simulert sitter man også igjen med den sanne koeffisientvektoren $\boldsymbol{\beta}$, som ble brukt til simuleringen av levetidene. Tabell 8.1 gir en oppsummering av de ulike simuleringsparameterne.

Parameter	Beskrivelse
n	Antall individer/observasjoner
p	Antall gener totalt
r	Antall gener som har betydning for hendelsestidene
$\boldsymbol{\beta}$	Koeffisientvektoren som ble brukt til å simulere data
s	Hvor stor innvirkning de r utvalgte genene skal ha på levetiden
z	Hvor stor andel av individene som sensureres
X	Matrise med kovariatverdier
Y	Matrise med levetider og sensurstatus (respons)

Tabell 8.1: De ulike parameterne som ble brukt til å simulere levetider.

8.1.2 Ekte data

Som ekte genekspresjonsdata ble det benyttet et datasett fra brystkreftpasienter. Dataene er tidligere beskrevet av Perou et. al.[41] og Sørlie et. al.[50]. Dataene er for 39 kvinner med lokalt framskreden brystkreft. Kvinnene ble behandlet med Doxorubicin monoterapi før operasjon, og gitt ulik behandling etter operasjonen. For hver pasient ble overlevelsestider registrert og brukt som hendelsestid i de videre studiene av dette datasettet. Median oppfølgingstid var 66 måneder, og dødsfall som skyldtes annet enn brystkreft ble registrert som sensurerte hendelser.

Datasettet inneholdt data for 1871 gener. For å plukke ut gener som er relevant for overlevelsen ble verktøyet *significance analysis of microarrays (SAM)* benyttet. Dette er en analytisk metode utviklet ved Stanford University Labs som korrelerer genekspresjonsdata til en rekke kliniske parametere, inkludert behandling, diagnosekriterier, overlevelsestider og tidstrender. Ved analyse av overlevelsesdata beregner SAM en score for hvert gen som måler styrken på korrelasjonen med overlevelsen. Denne scoren er den samme ML-score testobservatoren som kan beregnes i score-testen for Cox-modellen [28]. Negativ score indikerer at høy ekspresjon korrelerer med lenger overlevelse, mens positiv score indikerer at høy ekspresjon korrelerer med kortere overlevelse. SAM ble benyttet til å plukke ut 20 gener fra brystkreftdatasettet som korrelerte med overlevelsen. For å begrense antall gener ble disse satt sammen med 1000 andre tilfeldig valgte gener fra det samme datasettet, slik at det til sammen var 1020 gener i datasettet med ekte data som ble benyttet i de videre analysene.

8.2 Valg av estimeringsmetode

Ulike metoder for å finne et maksimum likelihood-estimat $\hat{\boldsymbol{\beta}}$ for den straffede Cox-modellen. Dette gjøres ved å optimere den straffede log-likelihooden til Cox-modellen fra avsnitt 4.7

$$l_{pen}(\boldsymbol{\beta}, \lambda) = \sum_{i=1}^{n} \delta_i(\boldsymbol{\beta}^T \mathbf{x}_i) - \sum_{i=1}^{n} \delta_i log \left\{ \sum_{j \in \mathcal{R}_{(t_i)}} exp(\boldsymbol{\beta}^T \mathbf{x}_j) \right\} - \sum_{j=1}^{p} \lambda_j \beta_j^2. \tag{8.1}$$

Estimeringsmetodene ble sammenliknet for å kunne bestemme hvilken som ville være best å benytte videre i arbeidet. Under evalueringen av metodene ble det lagt vekt på hvordan

de håndterte situasjoner med stort antall kovariater, spesielt når antall kovariater var mye mindre enn antall observasjoner. Det ble også lagt vekt på effektiviteten til metodene, ettersom man ofte må gjøre mange optimeringer under tilpasningen av en modell, blant annet ved modellseleksjon som vi skal se på senere i kapittelet.

Som alternativer til algoritmer for å kunne optimere den straffede log-likelihooden til Cox-modellen (8.1) ble en variant av Nelder-Mead simplex-algoritme og ulike varianter av Newtons optimeringsalgoritme testet ut. Begge algoritmene ble kort presentert i kapittel 5. Optimeringsalgoritmene ble testet ved å bruke den straffede log-likelihooden for Cox-modellen som funksjonen som skal optimeres på ulike simulerte datasett, der antall kovariater (gener) og antall observasjoner ble variert.

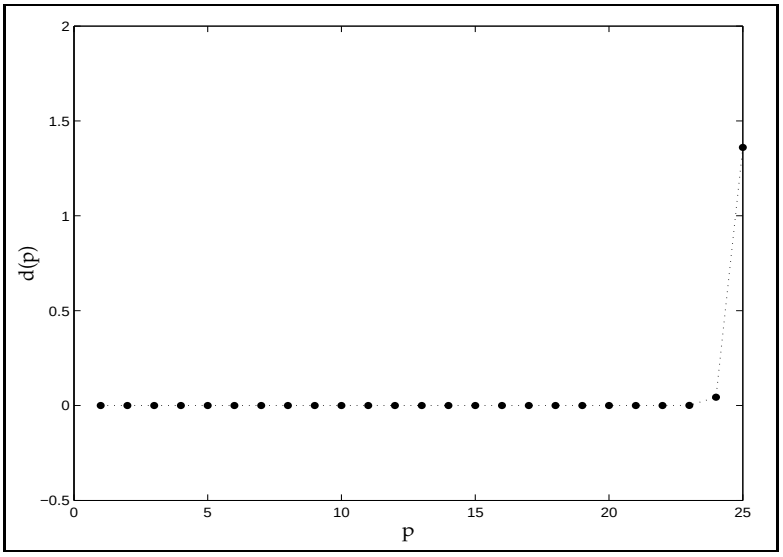
Som nevnt i kapittel 5 er en fordel ved simplex-algoritmen at den kun trenger å beregne selve funksjonsverdien i hver iterasjon, ingen informasjon om de deriverte. Dette medfører også en ulempe ved at metoden ofte konvergerer sakte. Som en implementasjon av simplex-metoden ble det i denne oppgaven benyttet funksjonen *fminsearch* som finnes i Matlab. Resultatene fra testene av denne (ikke presentert her) viste at den som forventet var svært lite effektiv, selv for et lite antall kovariater (antall gener $p = 20$). Denne optimeringsmetoden vil derfor ikke være brukbar i praksis til å løse den typen problemer vi ser på i denne oppgaven.

Kvasi-Newton algoritmer benytter seg i motsetning til simplex-algoritmen av både den førstederiverte og approksimasjoner til den annenderiverte i hver iterasjon for å optimere funksjonen. Flere varianter av slike Kvasi-Newton optimeringsalgoritmer ble også testet. En implementasjon av denne finnes i Matlab i funksjonen *fminunc* som følger med i optimerings-toolboxen (se dokumentasjonen til Matlab for detaljer). Denne benytter seg av BFGS-oppdatering av approksimasjonen til den andrederiverte, slik det ble beskrevet i avsnitt 5.3 og utfører linjesøk ved hjelp av kubisk og kvadratisk interpolasjon. Denne implementasjonen ble sammenliknet med en egen implementasjon av kvasi-Newton metoden laget i forbindelse med denne oppgaven. Implementasjonen bruker BFGS-oppdatering og linjesøkingsalgoritmen som ble presentert i avsnittene 5.3 og 5.4. Testene av disse to implementasjonene viste at de ga tilsvarende resultater, både når det kom til effektivitet og nøyaktighet, men *fminunc* var den mest robuste av de to. Dette skyldes trolig at den har en mer robust metode for å utføre linjesøk.

Funksjonen *coxph* i S-plus implementerer også en variant av Newton-Raphson optimeringsalgoritme for å tilpasse den straffede Cox-modellen. Det har ikke vært mulig å finne ut av detaljene rundt hvilken optimeringsalgoritme som benyttes, ettersom det står lite om det i dokumentasjonen til S-plus. Ved å benytte funksjonen *ridge* sammen med *coxph* kan man legge en ridge-straff på koeffisientene i Cox-modellen. Denne funksjonen ble testet på noen datasett og sammenliknet med resultater fra implementasjonen av kvasi-Newton metoden som ble laget for denne oppgaven. De to implementasjonene ble først sammenliknet på problemet (8.1) med $\lambda = 0$. Flere datasett med ulikt antall kovariater $p = 1, \ldots, 30$ ble simulert. For hvert slikt datasett estimerte *coxph* i S-plus koeffisientvektoren $\hat{\boldsymbol{\beta}}_{coxph}$ og kvasi-Newton implementasjonen koeffisientvektoren $\hat{\boldsymbol{\beta}}_{kN}$. Estimatene ble sammenliknet ved å regne ut differansen:

$$d(p) = \frac{||\hat{\boldsymbol{\beta}}_{coxph} - \hat{\boldsymbol{\beta}}_{kN}||^2}{p}, \tag{8.2}$$

for $p = 1, \ldots, 25$. Denne differansen er plottet i figur 8.1 for ulike verdier av p. Av figuren ser man at differansen som man kunne forvente er liten for få kovariater. Den øker derimot når



Figur 8.1: Differansen mellom estimert koeffisientvektor fra *coxph* i S-plus og den nye implementasjonen av Cox-modellen for ulikt antall kovariater p. Differansen mellom de to løsningene øker når vi får mange kovariater.

antall kovariater blir større enn 23. Når $p > 25$ klarer ikke lenger *coxph* å tilpasse modellen, og vi ser at differansen (8.2) øker kraftig rett før dette skjer.

Funksjonen *coxph* har som sagt også mulighet til å legge en ridge-straff på regresjonsparameterne, ved hjelp av funksjonen *ridge*. Funksjonen har en parameter, θ og ved å sette $\theta > 0$ medfører dette at man legger en ridgestraff på koeffisientene som tilsvarer $\theta/2$. Dette vil tilsvare å optimere (8.1) dersom man setter $\theta = 2\lambda$. For ulike verdier av λ ble resultatene sammenliknet ved å beregne (8.2). Så lenge $p \leq 25$ var resultatene tilnærmingsvis like. Det ble ingen resultater for større antall kovariater ettersom *coxph* ikke håndterer $p > 25$.

I de videre analysene i oppgaven vil både Matlabs *fminunc* og optimeringsalgoritmen implementert i forbindelse med oppgaven benyttes, ettersom resultatene i dette avsnittet viste at disse var mest effektive og klarte å håndtere mange kovariater. Fordelen med Matlabs funksjon er at den er robust, mens fordelen med den andre implementasjonen er at den ikke krever at man har installert optimerings-toolboxen i Matlab, noe som ikke følger med standard Matlab-pakken.

8.3 Estimering med fast valgt straffeparameter λ

Cox-modellen ble ved hjelp av den utvalgte optimeringsalgoritmen tilpasset på bakgrunn av datasett simulert med ulike parameterverdier. Hensikten var å finne ut hvordan modelltilpasningen påvirkes av endringer i simuleringsparameterne i tabell 8.1. Effekten av å endre på de ulike parameterne ble sammenliknet på bakgrunn av to evalueringskriterier, prediksjonsfeil og genseleksjon. For å kunne sammenlikne de ulike simulerte situasjonene på mest mulig likt grunnlag ble det for hvert datasett funnet en optimal straffeparameter λ^*, som ble brukt under

sammenlikningen. Vi skal først se på hvordan denne optimale straffeparameteren beregnes før evalueringskriteriene for de tilpassede modellene vil bli presentert.

8.3.1 Orakelestimatoren

Det var ønskelig å evaluere hvor godt den straffede Cox-modellen tilpasses datasett som simuleres med ulike verdier for simuleringsparameterne i tabell 8.1. For å kunne sammenlikne modelltilpasningen for hvert datasett er det ønskelig å finne den optimale straffede modellen i hver situasjon, for så å sammenlikne de estimerte modellene. En optimal straffet modell får man ved å finne en optimal straffeparameter λ^* for hver situasjon. Dette er den verdien av λ i den straffede log-likelihooden (8.1) som gir best tilpasning til den sanne modellen som dataene ble simulert fra.

Dersom man hadde valgt λ til en fast verdi $\lambda = \lambda_F$ som man så hadde brukt i alle situasjoner ville dette kunne resultert i at når man befant seg i situasjoner der den *egentlige* optimale λ^* tilfeldigvis er nær λ_F, dette kunne gitt tilsynelatende bedre resultater enn i andre situasjoner. Sammenlikninger av de samme situasjonene, men med en tilfeldig annen verdi for λ_F kunne gitt helt andre resultater. For eksempel vil det være naturlig å tro at i situasjoner med mange gener vil det være gunstig med en større straff en i situasjoner der vi har færre gener.

I avsnitt 3.12 så vi at for Cox-modellen kan risikoen til det i'te individ representeres ved den såkalte *risikoindeks*, definert som $\boldsymbol{\beta}^T\mathbf{x}_i$, der $\boldsymbol{\beta}$ er parametervektoren til modellen og $\mathbf{x}_i$ er kovariatvektoren til dette individet. I og med at denne oppgaven jobber med simulerte data er den sanne risikoen $\boldsymbol{\beta}^T\mathbf{x}_i$ for hvert individ i kjent. Ved å sammenlikne denne kjente risikoen med estimert risiko $\hat{\boldsymbol{\beta}}(\lambda)^T\mathbf{x}_i$ fra en straffet modell får man et mål på hvor godt den estimerte modellen passer med den sanne modellen. Dette kan gjøres ved å beregne gjennomsnittlig feil i predikert risiko (GFPR) over alle individer:

$$GFPR(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{\beta}^T\mathbf{x}_i - \hat{\boldsymbol{\beta}}(\lambda)^T\mathbf{x}_i)^2, \tag{8.3}$$

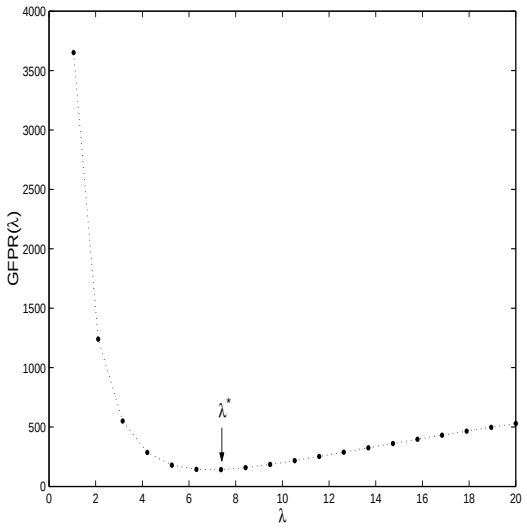
der n er antall individer i de simulerte dataene og $\hat{\boldsymbol{\beta}}(\lambda)$ er ML-estimatet for den den straffede Cox-modellen (8.1) med straffeparameter λ.

Ved å bruke GFPR som et mål for hvor godt den estimerte modellen stemmer med den sanne modellen kan en såkalt *orakelestimator* defineres ved uttrykket:

$$GFPR^* = \min_{\lambda} GFPR(\lambda). \tag{8.4}$$

Denne orakelestimatoren finner den verdien av straffeparameteren λ som er best i hver enkelt situasjon, det vil si for hvert datasett. Orakelestimatoren er altså den beste estimatoren som kan finnes med hensyn på prediksjon, ettersom straffen den velger er den som passer best med den sanne modellen. Se figur 8.2 for en illustrasjon av orakelestimatoren. Denne estimatoren finner den verdien λ^* som minimerer GFPR(λ), det vil si minimumspunktet på kurven.

For hvert simulerte datasett i dette avsnittet beregnes altså en λ^* som optimerer (8.4), og denne brukes til å tilpasse den straffede Cox-modellen for det samme datasettet. En slik tilpasning resulterer i et estimat $\hat{\boldsymbol{\beta}}(\lambda^*)$ og en optimal prediksjonsfeil $GFPR^*$. Ettersom alle



Figur 8.2: Plott over GFPR som funksjon av λ. Orakelestimatoren finner den verdien λ^* som minimerer denne funksjonen. I dette eksempelet blir λ=7.4. Eksempelet er for et ikke-ortogonalt datasett der n=50, r=50 og p=1000.

modellene tilpasses med en optimal straff vil de kunne sammenliknes etter de ulike evalueringskriteriene som vil bli presentert i neste avsnitt på et mest mulig likt grunnlag. Legg merke til at orakelestimatoren kun er mulig å beregne i oppsett der den sanne modellen er kjent. Hvordan man estimerer en ukjent straffeparameter uten bruk av den sanne modellen vil bli tatt opp i avsnitt 8.4.

8.3.2 Evalueringskriterier

To forskjellige evalueringskriterier ble benyttet for å sammenlikne prestasjonen til modeller estimert på bakgrunn av ulike datasett.

Prediksjon:
I det første kriteriet ble det målt hvor godt den estimerte straffede modellen fra (8.1) klarer å predikere risiko i forhold til sann risiko. Dette ble gjort ved å sammenlikne optimal prediksjon for hver modell, det vil si ved å se på $GFPR^*$ fra (8.4) for hver modell som estimeres.

Genseleksjon:
Det andre evalueringskriteriet som ble lagt til grunn for sammenlikning ser på hvor godt det er mulig å skille gener av betydning fra dem uten betydning. Et første alternativ for å gjøre dette er ved å se på absoluttverdiene til den estimerte koeffisientvektoren $\hat{\boldsymbol{\beta}}(\lambda^*)$. Ved å se hvor stor prosentandel av de r største verdiene som tilhørte de r genene som ble valgt til å ha betydning for levetiden i det simulerte datasettet, kan man få en ide om hvorvidt det er mulig å plukke ut genene med relevans for levetiden kun ved å se på absoluttverdien til de estimerte koeffisientene. Denne prosentandelen vil videre refereres til som *% top-gener*.

Et andre alternativ for å skille ut genene av betydning fra de uten betydning tar utgangspunkt

n	$\hat{\boldsymbol{\beta}}_B(\lambda^*)$ $\mathbf{P_{0.5}}$ $(\mathbf{P_{0.05}}, \mathbf{P_{0.95}})$	$\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ $\mathbf{P_{0.5}}$ $(\mathbf{P_{0.05}}, \mathbf{P_{0.95}})$	$\mathbf{GFPR}^*$	**% top-gener**	**#positive**
10	0.11 (-0.17, 0.24)	0.01 (-0.20, 0.20)	1.2903	20	17
50	0.58 (0.27, 0.73)	-0.03 (-0.33, 0.40)	1.4132	80	20
100	0.80 (0.67, 0.93)	-0.01 (-0.22, 0.25)	0.7988	100	11
200	0.92 (0.75, 1.24)	0.03 (-0.23, 0.22)	0.7880	100	10

Tabell 8.2: Tabellen viser median, 5- og 95-persentil for de estimerte koeffisientene delt inn i $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ og $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ (se teksten). Kolonnen $GFPR^*$ viser GFPR for orakelestimatoren for ulike verdier av n. Kolonnen merket med *% top-gener* angir hvor stor prosentandel av de r koeffisientene med størst absoluttverdi som faktisk tilsvarer de r genene som ble valgt til å ha betydning for levetiden under simuleringen. Kolonnen merket *# positive* viser antall gener valgt ut fra et plott av log-likelihoodverdier for stegvis utvidede modeller.

i orakelestimatet $\hat{\boldsymbol{\beta}}(\lambda)$ for den fulle modellen. På bakgrunn av dette estimatet beytter man så den metoden for å lage plott over log-likelihoodverdiene for modeller utvidet med en og en kovariat, som ble presentert i avsnitt 7.4.2. Antallet gener valgt ut fra slike log-likelihoodplott vil videre bli referert til som *#positive*.

I tillegg til disse evalueringskriteriene vil verdiene til de estimerte parameterkoeffisientene studeres. Disse vil kunne gi informasjon om hvor mye regulariseringen påvirker estimatene, i tillegg til hvor godt det er mulig å skille koeffisientene for gener med betydning for levetiden fra de uten betydning for levetiden. Definer $\hat{\boldsymbol{\beta}}_B(\lambda) = (\hat{\beta}_1(\lambda), \ldots, \hat{\beta}_r(\lambda))$, det vil si som subvektoren med de estimerte koeffisientene for de r kovariatene som har betydning for levetiden (se avsnitt 3.8 for detaljer). Tilsvarende defineres $\hat{\boldsymbol{\beta}}_{UB}(\lambda) = (\hat{\beta}_{r+1}(\lambda), \ldots, \hat{\beta}_p(\lambda))$ som koeffisientene til kovariatene uten betydning for levetiden. Dersom ikke annet er oppgitt er alle datasettene som er benyttet videre i oppgaven usensurerte og innvirkningen av genene med betydning for levetiden, s, er satt til 1.

8.3.3 Effekten av å endre antall gener og antall individer

Det ble først testet hvordan antall individer i forhold til antall gener totalt påvirker estimeringen av Cox-modellen. Modellen ble tilpasset med ulike ikke-ortogonale datasett (se avsnitt 8.1.1) for ulike kombinasjoner av antall individer og antall gener. Tabell 8.2 viser en oversikt over resultater fra en slik test når antall gener $p = 50$, antall gener av betydning $r = 10$ og antall observasjoner n ble variert.

Tabellen viser at når antall individer øker er det lettere å skille mellom koeffisientestimatene for genene av betydning og de uten betydning. Når $n = 100$, det vil si $n/p = 2$, blir ikke 5-persentilen til $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ og 95-persentilen $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ lenger overlappende, og det vil derfor være lettere å skille koeffisientene for gener med betydning og de uten betydning fra hverandre. Resultater for $GFPR^*$ viser at den gjennomsnittlige feilen i predikert risiko blir mindre ettersom antall individer øker i forhold til antall gener. Kolonnen %top-gener viser at når antall individer blir stort, det vil si at $n/p \geq 2$ er det mulig å plukke ut alle genene av betydning ved å se på de r største koeffisientverdiene. Fra kolonnen med # positive ser man at når det er få individer vil man ved bruk av denne metoden plukke ut for mange gener av betydning,

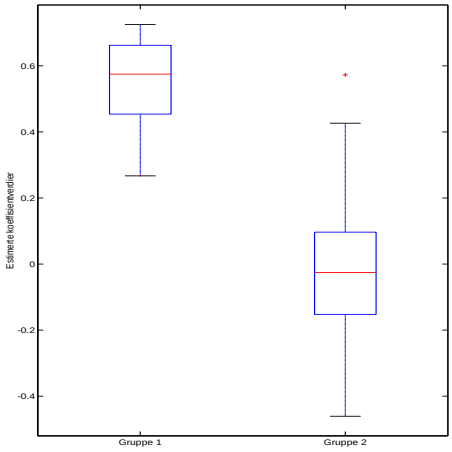
n	$\hat{\boldsymbol{\beta}}_B(\lambda^*)$ $\mathbf{P_{0.5}}$ $(\mathbf{P_{0.05}}, \mathbf{P_{0.95}})$	$\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ $\mathbf{P_{0.5}}$ $(\mathbf{P_{0.05}}, \mathbf{P_{0.95}})$	$\mathbf{GFPR}^*$	% top-gener
50	0.03 (-0.02, 0.10)	-0.00 (-0.07, 0.06)	5.25	10
100	0.09 (-0.00, 0.20)	-0.00 (-0.11, 0.11)	3.25	22
200	0.18 (0.04, 0.38)	-0.01 (-0.15, 0.15)	1.70	46

Tabell 8.3: Tabellen viser median ($P_{0.5}$) og 5- og 95-persentil ($P_{0.05}$ og $P_{0.95}$) for $\hat{\boldsymbol{\beta}}_B$ og $\hat{\boldsymbol{\beta}}_{UB}$ når antall gener $p = 1000$ og antall gener av betydning $r = 50$. Kolonnen $GFPR^*$ viser GFPR for orakelestimatoren for ulike verdier av n. Kolonnen % top-gener viser hvor stor prosentandel av de r koeffisientene med størst absoluttverdi som faktisk tilsvarer de r genene som ble valgt til å ha betydning for levetiden under simuleringen.

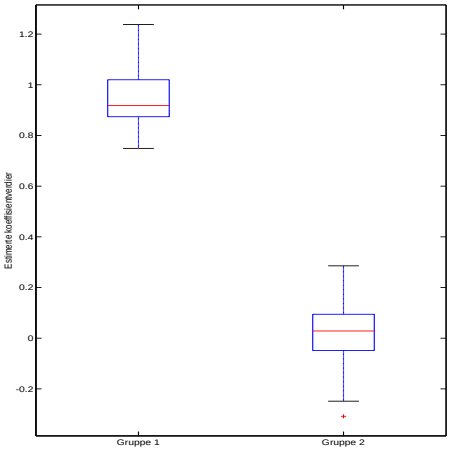
men når antall individer øker til over 50 kan man i dette tilfellet plukke ut korrekt antall kovariater. I figur 8.3 kan man se boksplott for $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ og $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ og de tilsvarende plottene for log-likelihoodverdiene i de stegvis utvidede modellene. I boksplottene ser man at når n er stor i forhold til p (figur 8.3(b)), er det lettere å skille mellom de estimerte koeffisientverdiene til gener av betydning (*Gruppe 1*) og gener uten betydning (*Gruppe 2*), enn når n er liten i forhold til p (figur 8.3(a)). I plottene av log-likelihoodverdiene for stegvis utvidede modeller er det mulig å se en knekk ved 10 kovariater når $n = 200$ (figur 8.3(d)). Når $n = 50$ (figur 8.3(c)) er det derimot vanskelig å se et klart knekkpunkt og det blir vanskelig å gi et godt estimat for antall gener av betydning ved bruk av denne metoden.

Tilsvarende tester ble utført for andre verdier av antall gener, der antall observasjoner ble variert. Et lavt antall gener $p \leq 200$ ga tilsvarende resultater som for de som ble vist for $p = 50$, det vil si at ettersom antall observasjoner økte var det lettere å separere gener av betydning fra de uten betydning, $GFPR^*$ økte og %top-gener økte mot 100%.

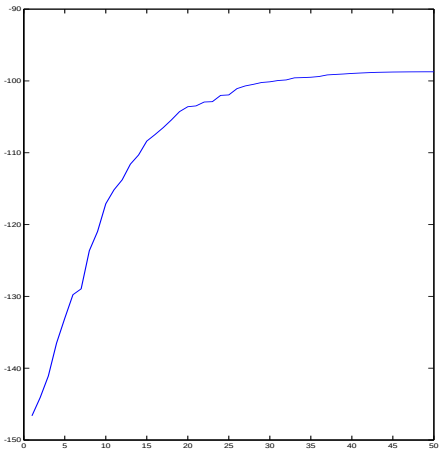
For større antall gener, $p = 200, 1000, 5000$, ble det testet kun for et mindre antall observasjoner i forhold til antall gener, ettersom estimeringstiden avhenger betraktelig av antall individer når antall kovariater blir stort, ettersom QR-algoritmen i avsnitt 5.5 ble benyttet. I disse situasjonene ble det derfor kun testet for $n/p \leq 1$, og ratioen ble gjort mindre ettersom antall gener økte. Det ble heller ikke regnet ut log-likelihoodverdier ved å ta med en og en kovariat som det ble gjort når antall kovariater var mindre, ettersom tiden det ville ta å estimere p slike modeller ville bli betraktelig mye større enn for liten p. Resultater for antall gener $p = 1000$ og antall individer $n = 50, 100, 200$ kan sees i tabell 8.3. Av tabellen ser vi at øvre og nedre persentil for henholdsvis $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ og $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ alltid er overlappende, så i situasjoner som dette blir det vanskelig å skille gener av betydning fra de uten betydning. Kolonnen $GFPR^*$ viser at prediksjonsfeilen for orakelestimatoren minker ettersom antall observasjoner blir større. Vi ser også at % top-gener er lav sammenliknet med når vi hadde mange observasjoner i forhold til antall gener, men prosentandelen øker ettersom antall observasjoner øker. Det er altså mulig å plukke ut flere gener av betydning med bruk av denne metoden ettersom antall observasjoner øker. (Andre verdier av p ga tilsvarende resultater (ikke vist)).



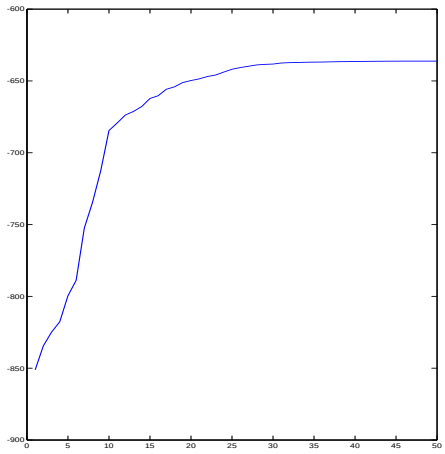
(a) Boksplott for $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ (Gruppe 1) og $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ (Gruppe 2) når n=50, p=50 og r=10.



(b) Boksplott for $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ (Gruppe 1) og $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ (Gruppe 2) når n=200, p=50 og r=10



(c) Log-likelihoodplott for stegvis utvidede modeller når n=50, p=50 og r=10. Log-likelihoodverdien, $l(\hat{\boldsymbol{\beta}})$, for de stegvis utvidede modellene er plottet mot antall gener inkludert i modellene.



(d) Log-likelihoodplott for stegvis utvidede modeller når n=200, p=50 og r=10. $l(\hat{\boldsymbol{\beta}})$ er plottet mot $\#gener$ inkludert i modellen. Log-likelihoodverdien, $l(\hat{\boldsymbol{\beta}})$, for de stegvis utvidede modellene er plottet mot antall gener inkludert i modellene.

Figur 8.3: Plott som viser noen av resultatene fra tabell 8.2

r/p	r	$GFPR^*$	% top-gener	# positive
0.1	5	0.78	80.0	25
0.2	10	1.01	100.0	10
0.4	20	1.45	80.0	27
0.6	30	1.63	90.0	35
0.8	40	4.04	90	35

Tabell 8.4: Tabell som viser utvalgte gener når $p = 50$ og $n = 50$ og r er variert. GFPR viser prediksjonsfeil for orakelestimatoren, % top-gener viser prosentandelen av de r største estimerte koeffisienene som tilhører gener med betydning for levetiden og # positive er antall gener av betydning plukket ut ved log-likelihoodplott.

r/p	r	$GFPR^*$	% top-gener	# positive
0.05	5	1.11	60.0	22
0.1	10	1.40	70.0	30
0.2	20	1.67	65.0	45
0.3	30	2.80	56.7	40
0.4	40	3.12	55.0	55
0.5	50	2.60	64.0	45
0.7	70	8.70	74.3	60
0.9	90	12.66	91.1	50

Tabell 8.5: Tabellen viser effekten av å endre r når p og n holdes fast. Resultatet er gitt for $p = 100$ og $n = 50$. GFPR viser prediksjonsfeil for orakelestimatoren, % top-gener viser prosentandelen av de r største estimerte koeffisienene som tilhører gener med betydning for levetiden og # positive er antall gener av betydning plukket ut ved log-likelihoodplott.

8.3.4 Effekten av å endre andel relevante gener for et fast antall observasjoner

Effekten av å endre andel gener av betydning for et fast antall observasjoner ble testet på tilsvarende måte som effekten av å endre antall observasjoner i avsnittet over. For ulike verdier av antall gener totalt ble andelen r/p, det vil si gener av betydning for levetiden i forhold til antall gener totalt, variert. Resultater ble sammenliknet på bakgrunn av $GFPR^*$, log-likelihoodplott for stegvis utvidede modeller og % top-gener på samme måte som vi så på i forrige avsnitt. Resultater for et utvalg slike tester finnes i tabellene 8.4, 8.5 og 8.6. Metoden med log-likelihoodplott for å finne # positve ble bare utført i de tilfellene der $p < 200$, ettersom det ville kreve betydelig beregningstid å gjøre dette når p er større. Av disse tre tabellene ser vi at prediksjonsfeilen øker ettersom andelen gener med betydning for levetiden øker i alle tilfeller. Økningen kan synes å være lineær opp til r/p er omlag 0.4 og deretter får den en kraftig økning . Vi ser av tabell 8.4 at når antall gener $p = 50$ varierer % top-gener forholdsvis lite ettersom andelen r/p endres. I tabell 8.5 varierer % top-gener rundt 60% før den stiger når r/p blir større enn 0.5, mens den øker mer eller mindre jevnt hele tiden i tabell 8.6 som viser for $p = 1000$.

Kolonnene med # positive i tabellene 8.4 og 8.5 viser at man ikke klarer å plukke ut riktig antall gener med relevans ved å bruke denne metoden. I begge tilfeller blir # positive for stort

r/p	r	$GFPR^*$	% top-gener
0.01	10	1.38	40.0
0.05	50	4.19	26.0
0.1	100	4.92	22.0
0.4	400	19.96	42.3
0.6	600	84.86	61.8
0.8	800	177.14	80.6
0.95	950	224.35	95.3

Tabell 8.6: Tabellen viser effekten av å endre r når p og n holdes fast. Resultatet er gitt for $p = 1000$ og $n = 100$. GFPR viser prediksjonsfeil for orakelestimatoren, % top-gener viser prosentandelen av de r største estimerte koeffisienene som tilhører gener med betydning for levetiden.

når $r/p < 0.5$ og for lite når $r/p \geq 0.5$.

8.3.5 Effekten av å endre korrelasjon mellom genene

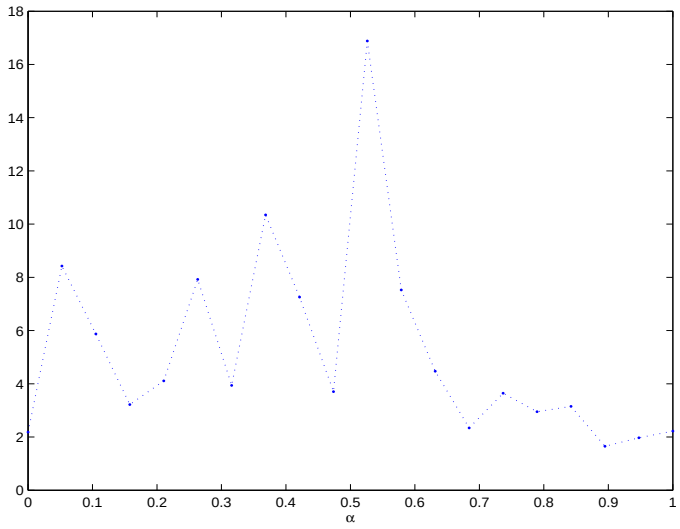
Effekten av å endre korrelasjonen mellom genene ble testet ved å bruke ulike andeler simulerte ortogonaliserte kovariater, X, og kovariater fra det ekte datasettet med brystkreftdata, $\tilde{X}$. Hensikten med dette var å se hvor stor effekt det hadde på estimatene $\hat{\boldsymbol{\beta}}(\lambda^*)$ å legge gradvis mer støy til de ortogonaliserte dataene, det vil si man tester hvor stor effekt korrelasjonen av genene har på den estimerte modellen. På denne måten vil man kunne se hvor langt man kan bevege seg bort fra den optimale ortogonale situasjonen, til en mer reell støyfull situasjon før det blir vanskelig å skille gener av betydning fra de uten betydning.

Testene ble utført ved å generere flere datasett

$$X^*_\alpha = \alpha\tilde{X} + (1-\alpha)X,$$

der $\tilde{X}$ er det simulerte ortogonale datasettet og X er datasettet med ekte genekspresjonsdata. Andelen α angir hvor stor andel av det simulerte datasettet som skal være med, og den ble variert mellom 0 og 1. For ulike verdier av α ble levetider simulert for datasettet X^*_α, og den straffede Cox-modellen ble tilpasset på bakgrunn av dette. For å evaluere hvor godt modellen ble estimert for ulike verdier av α ble $GFPR^*$ beregnet.

I figur 8.4 kan man se plott som viser $GFPR^*$ plottet mot verdier av α mellom 0 og 1, for et datasett med 39 personer, 1020 gener totalt der 20 av disse ble valgt til å ha betydning for levetidene. Tabell 8.7 viser en oversikt over de tilsvarende resultater for noen verdier av α. Vi ser av plottet og tabellen at det ser ut til at korrelasjonen i dette tilfellet ikke har så mye å si for prediksjonsfeilen, så lenge α er mindre enn 0.7. Vi ser av plottet at $GFPR^*$ varierer en god del når $\alpha < 0.7$, men stabiliserer seg på et litt lavere nivå ettersom korrelasjonen mellom genene øker, det vil si når $\alpha > 0.7$. Vi ser også fra tabellen at median verdi for $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ øker ettersom korrelasjonen mellom genene blir mindre. Vi ser også at 5-persentilen og 95-persentilen for henholdsvis $\hat{\boldsymbol{\beta}}_B(\lambda^*)$ og $\hat{\boldsymbol{\beta}}_{UB}(\lambda^*)$ er overlappende i alle tilfeller unntatt når all korrelasjon er fjernet. Likevel øker % top-gener jo mindre korrelasjonen blir, så det



Figur 8.4: Plott av gjennomsnittlig feil i predikert risiko fra tilpasninger av Cox modell med ulik andel korrelasjon mellom genene. Plottet er beregnet for et datasett med $n = 39$, $p = 1020$ og $r = 20$, beskrevet nærmere i teksten.

er lettere å plukke ut gener med betydning for levetidene jo lavere korrelasjon det er mellom genene.

8.3.6 Effekten av å endre andelen sensurerte observasjoner

Til slutt ble effekten av å endre graden av sensur i dataene testet. Parameteren z som angir andelen ikke-sensurerte observasjoner ble variert mellom 0.1 og 1 for datasett simulert med ulike parameterverdier. Resultater for to slike tester kan sees i tabell 8.8 og 8.9. Tabell 8.8 viser resultater $n = 100$, $p = 50$ og $r = 10$. Fra tabellen ser man at prediksjonsfeilen, $GFPR^*$, blir mindre jo lavere sensurandelen er. Når andelen ikke-sensurerte blir større enn 0.5 er det også mulig å plukke ut alle gener med relevans ved å se på % top-gener. Det er

$\boldsymbol{\alpha}$	$\hat{\boldsymbol{\beta}}_B$ $\mathbf{P_{0.5}}$ $(\mathbf{P_{0.05}}, \mathbf{P_{0.95}})$	$\hat{\boldsymbol{\beta}}_{UB}$ $\mathbf{P_{0.5}}$ $(\mathbf{P_{0.05}}, \mathbf{P_{0.95}})$	**GFPR***	**% top-gener**
0	0.05 (0.01, 0.18)	0.01 (-0.05, 0.07)	2.18	3.0
0.26	0.05 (0.00, 0.18)	0.01 (-0.06, 0.08)	7.92	3.0
0.53	0.02 (-0.00, 0.05)	0.00 (-0.02, 0.02)	16.89	5.0
0.74	0.21 (0.06, 0.46)	0.00 (-0.09, 0.09)	3.64	14.0
1.00	0.78 (0.29, 1.29)	0.00 (-0.01, 0.01)	2.22	20.0

Tabell 8.7: Tabellen viser median ($P_{0.5}$) og 5- og 95-persentil ($P_{0.05}$ og $P_{0.95}$) for $\hat{\boldsymbol{\beta}}_B$ og $\hat{\boldsymbol{\beta}}_{UB}$ for ulik korrelasjon mellom genene. Kolonnen $GFPR^*$ viser GFPR for orakelestimatoren for ulike verdier av α. Kolonnen % top-gener viser hvor stor prosentandel av de r koeffisientene med størst absoluttverdi som faktisk tilsvarer de r genene som ble valgt til å ha betydning for levetiden under simuleringen.

z	$GFPR^*$	% top-gener	# positive
0.1	4.84	60.0	19
0.2	3.61	80.0	22
0.3	1.93	90.0	20
0.4	1.79	90.0	13
0.5	1.38	90.0	17
0.6	1.15	100	16
0.7	0.96	100	15
0.8	0.92	100	10
0.9	0.84	100	10
1	0.78	100	10

Tabell 8.8: Tabellen viser effekten av å endre andelen sensur i dataene. Kolonnen $GFPR^*$ viser GFPR for orakelestimatoren for ulike verdier av z. Kolonnen merket med *% top-gener* angir hvor stor prosentandel av de r koeffisientene med størst absoluttverdi som faktisk tilsvarer de r genene som ble valgt til å ha betydning for levetiden under simuleringen. Kolonnen merket *# positive* viser antall gener valgt ut fra et plott av log-likelihoodverdier for stegvis utvidede modeller. Resultatene er vist for $n = 100$, $p = 50$ og $r = 10$.

z	$GFPR^*$	% top-gener
0.1	61.1	18
0.2	44.2	21
0.3	34.8	23
0.4	26.3	24
0.5	20.0	19
0.6	16.0	20
0.7	14.1	22
0.8	12.4	23
0.9	9.9	23
1	7.1	23

Tabell 8.9: Tabellen viser effekten av å endre andelen sensur i dataene. Kolonnen $GFPR^*$ viser GFPR for orakelestimatoren for ulike verdier av z. Kolonnen merket med *% top-gener* angir hvor stor prosentandel av de r koeffisientene med størst absoluttverdi som faktisk tilsvarer de r genene som ble valgt til å ha betydning for levetiden under simuleringen. Kolonnen merket *# positive* viser antall gener valgt ut fra et plott av log-likelihoodverdier for stegvis utvidede modeller. Resultatene er gitt for $n = 100$, $p = 1000$ og $r = 100$.

derimot ikke mulig å plukke ut riktig # positive før andelen ikke-sensurerte er større enn 0.7. Resultatene i tabell 8.9 er gitt for $n = 100$, $p = 1000$ og $r = 100$. I denne situasjonen minker prediksjonsfeilen ettersom andelen ikke-sensurerte observasjoner øker, tilsvarende som vi så i tabell 8.8. Tabellen viser også at en forholdsvis stor sensurandel, det vil si at $z < 0.4$, til en viss grad har innvirkning på antall gener av betydning det er mulig å plukke ut. En mindre sensurandel, $z \geq 0.4$, har lite eller ingen betydning for hvor mange gener av betydning man klarer å plukke ut.

8.4 Modellseleksjon - Estimering av straffeparameter λ

Til nå har alle estimeringer blitt gjort med straffeparameteren λ satt til en fast verdi λ^*. Denne verdien har blitt valgt ved å bruke orakelestimatoren (8.4) som for ethvert datasett gir den beste straffeparameteren med hensyn på predikert risiko. Orakelestimatoren gjør dette ved å benytte seg av at vi kjenner den sanne modellen for de simulerte dataene. For ekte genekspresjonsdata er den sanne modellen ukjent, og man må derfor prøve å estimere straffeparameteren λ. Dette gjøres ved å benytte metoder for modellseleksjon, som finner den beste modellen ved å variere på straffeparameteren λ som. Vi skal i dette avsnittet se på resultater fra metodene for modellseleksjon presentert i kapittel 6. Resultatene fra metodene vil bli evaluert blant annet etter kriterier for prediksjon og genseleksjon.

8.4.1 Test av kryssvalideringsmetoder

De tre variantene av kryssvalidering (CV) for en straffet Cox-modell som ble presentert i kapittel 6 ble implementert og det ble valgt å teste ut metodene på ulike simulerte datasett, både ortogonale og ikke-ortogonale. Fordelen ved å bruke simulerte datasett er som nevnt tidligere at den sanne modellen med parameter $\boldsymbol{\beta}$ da er kjent, slik at det vil være mulig å se hvor godt de ulike metodene estimerer λ i forhold til den optimale λ som finnes av orakelestimatoren.

Metodene ble testet i ulike situasjoner der antall observasjoner n, antall gener p og antall gener av betydning r ble variert. For hver situasjon, det vil si for hvert valg av n, p og r ble det kjørt et visst antall kryssvalideringer med hver CV-variant for å kunne si noe om reproduserbarhet av resultatene i hver situasjon. Hver kryssvalidering gikk ut på å først simulere et treningsdatasett med de valgte parameterne. Deretter ble de tre CV-variantene trenet på dette datasettet, noe som resulterer i et estimat for optimal straff, $\hat{\lambda}$, og et straffet estimat $\hat{\boldsymbol{\beta}}(\hat{\lambda})$ for hver av metodene. Videre ble disse estimatene brukt til å evaluere de tre variantene etter to kriterier:

- Log-likelihooden for hver av de estimerte modellene ble regnet ut. Den av de tre CV-variantene som gir størst log-likelihoodverdi for det endelige estimatet $\hat{\boldsymbol{\beta}}(\hat{\lambda})$ beregnet for testsettet, vil være den modellen som passer best med denne typen data (se appendiks A).
- Prediksjon av risiko for nye individer ble studert. Dette ble gjort ver å generere et testdatasett på tilsvarende måte som treningssettet, slik at testsettet består av samme type data, bare med nye individer. Dette treningsdatasettet består derimot av et stort

Metode	$\hat{\lambda}$	$l(\hat{\boldsymbol{\beta}}(\hat{\lambda}))$	GFPR($\hat{\lambda}$)
CV-variant 1	2.90	-245.69	2.29
CV-variant 2	14.95	-277.54	4.00
CV-variant 3	3.65	-249.48	2.27
Orakel	2.03	-244.20	1.82

Tabell 8.10: Tabellen viser gjennomsnittlige verdier over 10 kryssvalideringer for $n = 100$, $p = 50$ og $r = 10$. Tabellen viser estimert straffeparameter, $\hat{\lambda}$, log-likelihoodverdien for den estimerte parametervektoren, $l(\hat{\boldsymbol{\beta}}(\hat{\lambda}))$, og gjennomsnittlig prediksjonsfeil for de ulike metodene, GFPR($\hat{\lambda}$).

antall observasjoner slik at det kan brukes til å si noe om gjennomsnittlig prediksjonsfeil for nye individer. Prediksjonsfeil $GFPR(\hat{\lambda})$ fra (8.3) ble beregnet for hver av de tre CV-variantene. Ettersom også testdataene er simulerte kan man regne ut $GFPR^*$ for datasettet ved å bruke orakelestimatoren. $GFPR(\hat{\lambda})$ for hver av de tre CV-metodene ble så sammenliknet innbyrdes og med $GFPR^*$. Merk at egenskaper ved orakelestimatoren gjør at denne estimatoren alltid vil gi den beste verdien for $GFPR$ vi klarer å finne. Den av metodene som gir $GFPR(\hat{\lambda})$ nærmest orakelestimatoren, vil være den beste modellen for prediksjon av risiko for nye individer.

I tillegg til disse evalueringskriteriene ble de estimerte parameterne, $\hat{\lambda}$ og $\hat{\boldsymbol{\beta}}(\hat{\lambda})$, fra hver CV-variant sammenliknet direkte for hver kryssvalidering. Hensikten var å avdekke likheter mellom metodene i praksis og eventuelt se om disse kunne gjenspeile noen av de teoretiske likhetene påpekt i avsnitt 6.6.

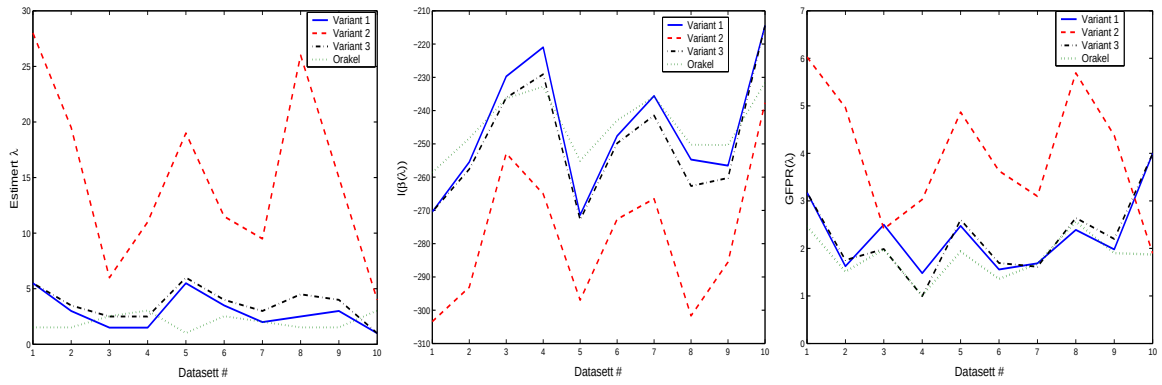
Tester med få gener og mange observasjoner

I første omgang ble datasettene generert med forholdsvis få kovariater, det vil si $p < 100$ og relativt mange observasjoner i forhold til antall kovariater, det vil si $n/p > 0.5$. Vi definerer $\hat{\boldsymbol{\beta}}_B(\lambda)$ og $\hat{\boldsymbol{\beta}}_{UB}(\lambda)$ som tidligere i kapittelet.

Tabellene 8.10 og 8.11 gir et eksempel på resultater fra 10 kryssvalideringer med de tre CV-variantene for ikke-ortogonale datasett med 100 observasjoner og 50 gener, der 10 av disse genene hadde betydning for levetiden. Resultatene er gitt som gjennomsnittet over de 10 kryssvalideringene. Vi ser av tabell 8.10 at i denne situasjonen gir CV-variant 1 og 3 temmelig like resultater for både log-likelihood og prediksjonsfeil mens CV-variant 2 gir dårligst resultat for begge kriteriene. Resultatene er lettere å sammenlikne dersom man ser på plott over estimert optimal straffeparameter $\hat{\lambda}$, log-likelihoodverdier for $\hat{\boldsymbol{\beta}}(\hat{\lambda})$ og gjennomsnittlig feil i predikert risiko i forhold til de tilsvarende estimatene fra orakelestimatoren. I figur 8.5 er $\hat{\lambda}$, $l(\hat{\boldsymbol{\beta}}(\hat{\lambda}))$ og $GFPR(\hat{\lambda})$ for hver av de 10 ulike treningssettene brukt i kryssvalideringene plottet. Av figuren ser man at CV-variant 1 og 3 velger liknende verdier for $\hat{\lambda}$ i hver kryssvalidering, noe som medfører at de også får liknende verdier for log-likelihood og prediksjonsfeil. CV-variant 2 gir større verdier for $\hat{\lambda}$ i alle de 10 kryssvalideringene, det vil si at CV-variant 2 velger å straffe koeffisientene mer enn de to andre CV-variantene. Dette gjenspeiles også i estimatene for median og 5- og 95-persentiler til koeffisientverdiene for $\hat{\boldsymbol{\beta}}_B$ og $\hat{\boldsymbol{\beta}}_{UB}$ som finnes

Metode	$\hat{\boldsymbol{\beta}}_B(\hat{\lambda})$	$\hat{\boldsymbol{\beta}}_B(\hat{\lambda})$	$\hat{\boldsymbol{\beta}}_{UB}(\hat{\lambda})$	$\hat{\boldsymbol{\beta}}_{UB}(\hat{\lambda})$
CV-variant 1	0.83	(0.53, 1.08)	0.00	(-0.28, 0.26)
CV-variant 2	0.42	(0.25, 0.57)	-0.00	(-0.13, 0.15)
CV-variant 3	0.76	(0.48, 0.99)	0.00	(-0.25,0.24)
Orakel	0.87	(0.56, 1.12)	0.00	(-0.30, 0.29)

Tabell 8.11: Tabellen viser gjennomsnittlige verdier for median, 5- og 95-persentil for de estimerte koeffisientvektorene over 10 kryssvalideringer, for $n = 100$, $p = 50$ og $r = 10$..



Figur 8.5: Plott for resultater av 10 kryssvalideringer der $n = 100$, $p = 50$ og $r = 10$. Plottet til venstre viser estimerte straffeparametere for hver av de 10 kryssvalideringene, plottet i midten viser log-likelihoodverdier for hver av kryssvalideringene og plottet til høyre viser gjennomsnittlig prediksjonsfeil for hver av de 10 kryssvalideringene.

i tabell 8.11. De estimerte koeffisientvektorene for CV-variant 2 er straffet mer, og dermed blir estimatet for median av $\hat{\boldsymbol{\beta}}(\hat{\lambda})$ lavere enn for de to andre CV-variantene. CV-variant 1 og 3 gir et bedre estimat enn CV-variant 2 for den sanne parameteren $\boldsymbol{\beta}$, som i dette tilfellet hadde verdien 1 for de r første komponentene og 0 for resten.

Tilsvarende kryssvalideringer ble utført for flere ulike verdier av n, r og $p < 100$ for både ortogonale og ikke-ortogonale datasett. Spesifikke resultater fra dette er ikke gjengitt her, men de viste stort sett det samme som resultatene presentert over. Generelt ga CV-variant 1 og 3 rimelig like resultater, både for $\hat{\lambda}$, log-likelihood og prediksjon i alle testene som ble utført. CV-variant 2 var alltid den dårligste både etter log-likelihood og prediksjonskriteriet. I tillegg valgte CV-variant 2 en større straff enn de to andre variantene i stort sett alle tilfellene, som medførte at parameterestimatene ble mer straffet.

Tester med mange gener og få observasjoner

De ulike variantene av kryssvalidering ble også testet i situasjoner der antall gener var mye større enn antall observerte individer. I disse situasjonene oppsto det problemer i noen av metodene med å finne en optimal straffeparameter. I avsnitt 6.5 så vi at CV-variant 1 og 2 finner optimal straff ved å optimere kryssvalideringskriteriene $CV_1(\lambda)$ (6.5) og $CV_2(\lambda)$ (6.8) henholdsvis. I situasjonene der antall gener var større enn antall observasjoner ble derimot ingen av disse kriteriene optimert for noen verdi av λ, men de fortsatte å øke inntil λ ble uendelig stor. Det var derfor ikke mulig å teste hvordan disse metodene presterer i slike situasjoner.

CV-variant 3 fant derimot et estimat for optimal λ i alle situasjoner med mange gener og få observasjoner som ble testet her. Denne metoden ble derfor testet videre. Ettersom det nå ikke lenger er mulig å sammenlikne hvordan denne metoden presterer i forhold til de andre kryssvalideringsmetodene vil den kun bli sammenliknet med orakelestimatoren. Måten dette vil bli gjort på er å regne ut forholdet mellom $GFPR(\hat{\lambda})$ for CV-variant 3 og $GFPR^*$ for orakelestimatoren. Dette forholdet vil bli regnet ut ved å se på uttrykket:

$$Err = log_2[\frac{GFPR(\hat{\lambda})}{GFPR^*}]. \tag{8.5}$$

En fordel ved å velge å se på log_2 av forholdet er at man får da en mer intitiv skala å forholde seg til, der $Err = 0$ angir at metodene er like gode og Err øker ettersom prediksjonen for CV-variant 3 blir dårligere i forhold til orakelestimatoren. Dersom CV-variant 3 gir dobbelt så stor feil vil Err bli 1, gir den fire ganger så stor feil vil Err bli 2 og så videre.

CV-variant 3 ble testet ut i en rekke situasjoner på samme måte som beskrevet over, der det for hver situasjon ble kjørt flere kryssvalideringer for å få sikrere resultater. I dette tilfellet ble det, i motsetning til når antall kovariater var lite, benyttet flere små testsett for hver kryssvalidering istedenfor ett stort. Dette ble valgt å gjøre ettersom det å benytte ett stort testsett ville kreve betydelig mer beregningstid nå som antall kovariater er blitt stort, enn når vi så på et lite antall kovariater. Her ble det valgt at hvert testsett skulle ha samme antall observasjoner som treningssettet. Resultatene for prediksjonsfeil for testsett er gitt som gjennomsnittet av prediksjonsfeilen over disse små testsettene.

Parameterverdier	$Err_{trening}$	Err_{test}
p=500, n=50	0.9451	2.5432
p=500, n=100	0.6700	2.5015
p=1000, n=50	3.3767	4.0802
p=1000, n=100	2.7526	3.7205

Tabell 8.12: Tabellen viser gjennomsnittlige resultater for flere situasjoner der antall gener og antall gener av betydning er variert som angitt i kolonnen med parameterverdier. $Err_{trening}$ er forskjell i treningsfeil for CV-variant 3 og orakelestimatoren. Err_{test} er tilsvarende forskjell i testfeil.

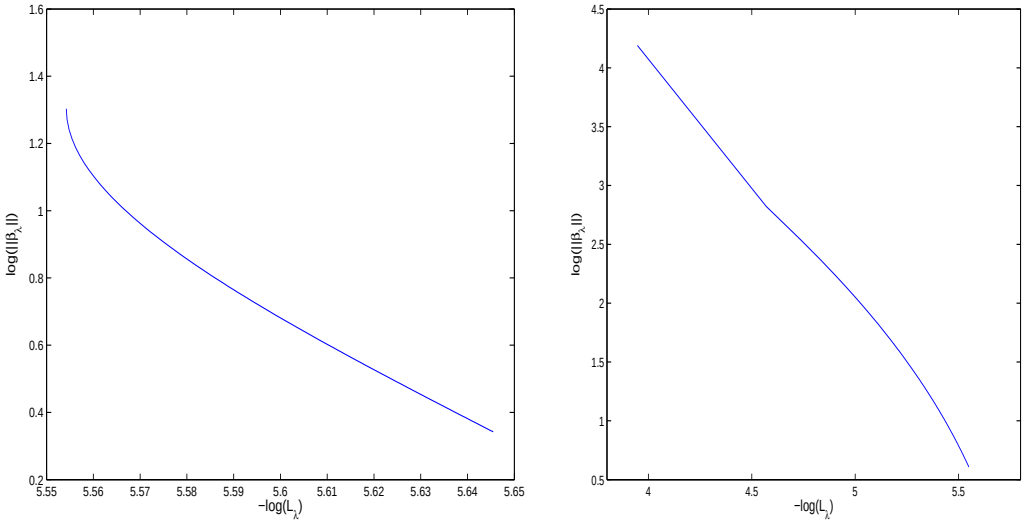
For hver av kryssvailderingene ble uttrykket 8.5 også beregnet. Err ble beregnet både for treningssettet som ble brukt under kryssvalideringen og som gjennomsnittet over testsettene. Tabellen 8.12 viser resultater for fire situasjoner, der n og p er variert. Hver situasjon er testet med 5 kryssvalideringer og for hver kryssvalidering er det brukt 5 små testsett for å beregne testfeil. Vi ser at i alle disse situasjonene er feilen for treningssettet som forventet mindre enn for de uavhengige testsettene. Antall gener som inkluderes i modellen har også effekt på prediksjonsfeilen. Dersom antall kovariater dobles fra 500 til 1000 blir prediksjonsfeilen omlag dobbelt så stor i forhold til orakelestimatoren som ved bare 500 kovariater. Dette er til tross for at antall observasjoner i forhold til antall kovariater er likt i begge tilfellene.

8.4.2 Test av L-kurve kriteriet

L-kurve kriteriet fra kapittel 6 ble også testet på flere ulike simulerte datasett. L-kurvene plotter som sagt i avsnitt 6.7 logaritmen av normen til den straffede parametervektoren $\hat{\boldsymbol{\beta}}(\lambda)$ mot logaritmen til den ustraffede log-likelihooden $log(l(\hat{\boldsymbol{\beta}}(\lambda)))$ for økende verdier av λ i et intervall $[\lambda_{min}, \lambda_{max}]$. En slik kurve er forventet å få en L-liknende form, og det er ønskelig å finne punktet med høyest krumning på kurven som tilsvarer at forholdet mellom straffingen av $\boldsymbol{\beta}$ og log-likelihooden er optimal.

Resultater for to L-kurver beregnet for datasett med få antall kovariater, $p = 50$, i forhold til antall observasjoner, $n = 100$, kan sees i figur 8.6. Fem av genene er valgt til å ha betydning for levetiden. Plottet til venstre viser når betydningen $s = 1$, mens plottet til høyre viser når betydningen $s = 10$. Vi ser at kurven endrer form ettersom verdien, s, av de ekte koeffisientene til genene med betydning øker. Plottet til venstre har krumning til riktig retning, tilsvarende en slags L-form, mens plottet til høyre nærmest blir en rett linje, før den krummer feil vei ettersom λ øker. I figur 8.7 er L-kurver plottet for de samme verdiene som for figur 8.6, bortsett fra at antall gener av betydning er endret til $r = 45$. Igjen får kurven til venstre der betydningen av de relevante genene er satt til $s = 1$, forventet krumningsform, mens kurven i plottet til høyre krummer klart i en annen retning enn det man ville forvente.

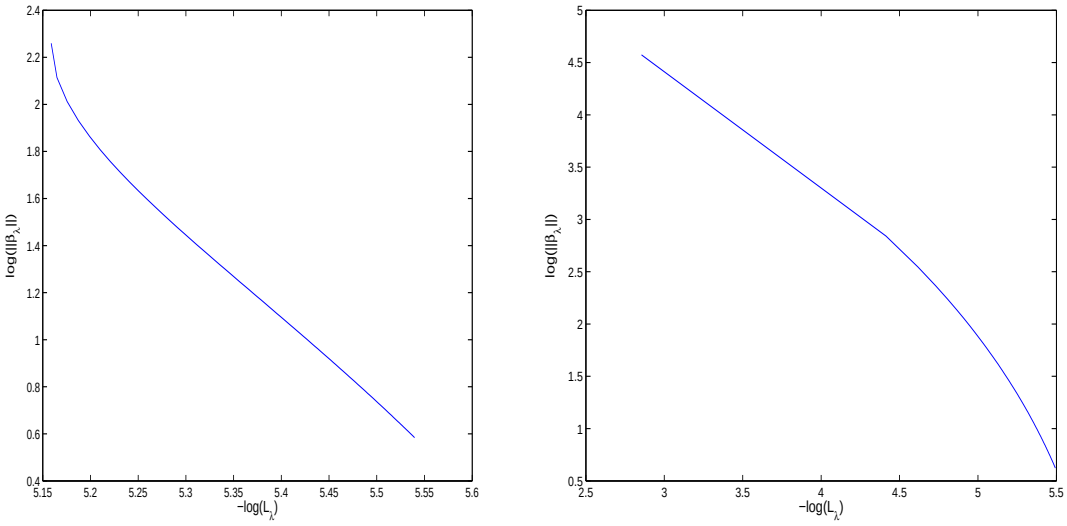
Det ble også testet å plotte L-kurver for datasett med et stort antall kovariater i forhold til antall observasjoner, $n = 100$. Plottene i figur 8.8 viser eksempler på to slike tester. Plottet til venstre er for antall gener $p = 1000$ og antall gener med betydning $r = 50$, og betydningen s er valgt til 1. Plottet til høyre er for $p = 2000$ der antall gener med betydning $r = 1500$ og betydningen $s = 5$. Heller ikke disse kurvene får den karakteristiske L-formen som man kunne



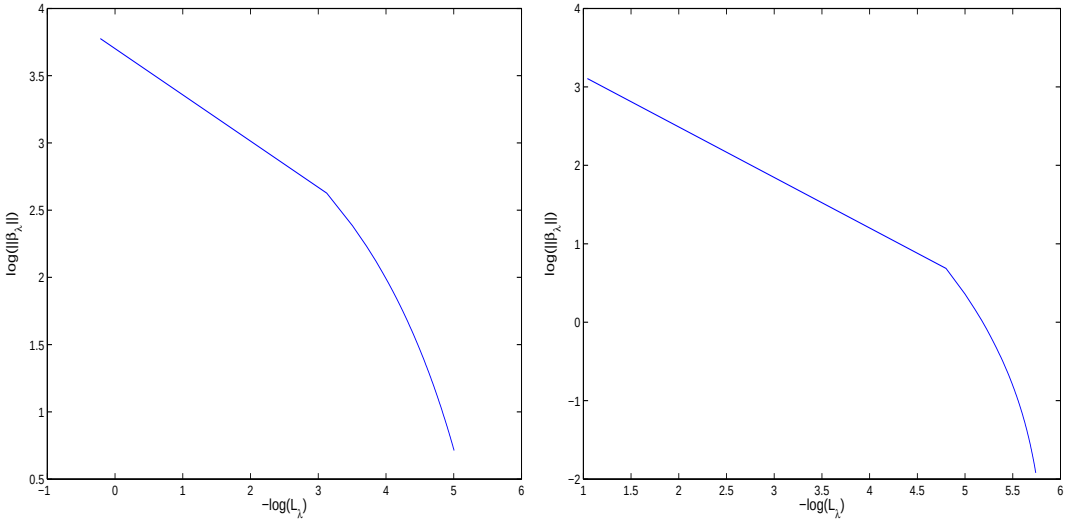
Figur 8.6: L-kurver for datasett med relativt mange observasjoner, $n = 100$, og relativt få gener, $p = 50$, der 5 av disse genene er valgt til å ha betydning for levetiden. I det venstre plottet er betydningen lav, $s = 1$, mens i det høyre plottet et betydningen høy, s=10.

forvente, men får heller krumningen i speilvendt retning.

Ideen med L-kurvene er at man kan finne den verdien av λ som ligger der buen har størst krumning ved å studere disse kurvene. Ved å regne ut $GFPR$ for denne verdien for λ kan denne sammenliknes med $GFPR^*$ for orakelestimatoren 8.4 på samme måte som vi gjorde for CV-variant 3 i forrige avsnitt ved å bruke $Err(8.5)$. Dette gir et mål på hvor godt denne metoden fungerer, og kunne sammenliknes med for eksempel de ulike kryssvalideringsvariantene. Ettersom disse plottene i dette avsnittet viste så uventede resultater ble det derimot ikke gjort forsøk på å finne optimalt krumningspunkt på disse kurvene, for å bestemme optimal straffeparameter. Det ble isteden jobbet med å forstå hvorfor kurvene fikk denne uventede formen. Teoretiske betraktninger rundt metoden og videre litteraturstudier ga ingen forklaring på dette fenomenet. Problemstillingen vil derfor bli tatt opp videre i diskusjonen i kapittel 9.



Figur 8.7: L-kurve for datasett med relativt mange observasjoner n=100, og relativt få gener p=50, der 45 av disse genene er valgt til å ha betydning for levetiden. I det venstre plottet er betydningen lav, s=1, mens i det høyre plottet et betydningen høy, s=10.



Figur 8.8: L-kurve for datasett med mange kovariater og få observasjoner (n=100). Plottet til venstre er for 1000 gener, der 50 av disse har betydning for levetiden (s=1). Plottet til høyre er for 2000 gener der 1500 av disse har relativt stor betydning for levetiden (s=5).

Kapittel 9

Diskusjon og videre arbeid

9.1 Oppsummering

Vi har sett på bruk av Cox-proporsjonale hasardmodell til å modellere overlevelse ut fra genekspresjonsdata fra mikroarrayer. Når genekspresjonsdataene er høydimensjonale, det vil si at det er mange flere gener enn observasjoner, er det ikke mulig å gjennomføre en klassisk tilpasning av Cox-modellen. Som en løsning på dette problemet ble en regularisert versjon av modellen benyttet, der log-likelihooden ble tilpasset med en L_2-type straff. Den straffede Cox-modellen ble i hovedsak testet ut på simulerte datasett. Hensikten med dette var for det første å se hvordan ulike parametere påvirket tilpasningen av modellen. De estimerte modellene ble evaluert etter ulike kriterier, blant annet etter hvor godt de predikerer risiko for nye individer og hvor lett det var å gjøre genseleksjon på bakgrunn av modellene. Ulike metoder for modellseleksjon ble testet ut og vurdert etter kriterier for prediksjon og genseleksjon. I tillegg til å teste eksisterende modellseleksjonsmetoder for Cox-modellen ble det lagt fram forslag til to metoder for modellseleksjon som ikke tidligere har vært brukt i forbindelse med denne modellen.

9.2 Diskusjon

9.2.1 Valg av modell

Cox proporsjonale hasardmodell er en hyppig brukt modell innen overlevelsesanalyse. Dette kan dels forklares ved at modellen har vist sin nytte i mange sammenhenger. I tillegg leder en slik modell til enkle uttrykk både for log-likelihood og de deriverte av log-likelihooden. Modellens popularitet har gjort at det er opparbeidet mye forståelse rundt modellen, og den har et stort metode- og begrepsapparat knyttet til seg. Det finnes en rekke andre modeller for analyse av overlevelsesdata som også kunne vært interessante å teste ut i sammenheng med genekspresjonsdata, men det har falt utenfor rammen av denne oppgaven å omtale disse.

9.2.2 Valg av regulariseringsmetode

En ridge-type straff ble valgt som regulariseringsmetode for Cox-modellen. Denne typen straff ble valgt fordi ridge regresjon er en enkel og vanlig form for regularisering som har vist seg nyttig og virkningsfull i tilsvarende problemstillinger innen andre fagfelt. Det ble derfor ansett som interessant å forsøke denne typen regularisering på problemene som har blitt studert i denne oppgaven. Ridge-type straff har en rekke gode egenskaper, blant annet har den en enkel krympingsstruktur, noe som gjør det mulig å gjøre en variabelseleksjon på bakgrunn av de regulariserte parameterestimatene. Det kunne vært interessant å studere Cox-modellen i forbindelse med andre regulariseringsmetoder som for eksempel PLS eller PCR. Det er allerede gjort en del tilnærminger til dette i andre studier, og noen av disse ble kort presentert i avsnitt 4.8.

9.2.3 Valg av algoritme for estimering av modellen

Tre algoritmer for tilpasning av den straffede Cox-modellen ble testet ut. Algoritmene ble vurdert etter hvor godt de håndterte situasjoner der datasettet har mange kovariater og få observasjoner, slik det er vanlig for mikroarray genekspresjonsdata. Resultatene viste at algoritmen som tilpasser Cox-modellen i den populære programpakken S-plus ikke klarte å håndtere mer enn 25 kovariater, til tross for at antall observasjoner var langt større enn 25. Dette viser at denne algoritmen i S-plus ikke egner seg til analyse av genekspresjonsdata, og det var følgelig et klart behov for alternative implementasjoner for tilpasninger av Cox-modellen.

Tilsvarende tester for andre optimeringsalgoritmer resulterte i at en iterativ Kvasi-Newton algoritme ble valgt. Denne viste seg å være rask og robust, selv på store problemer med mange gener og mange individer. En effektivisering av optimeringsproblemet ble funnet ved å bruke en QR-dekomposisjon av kovariatmatrisen, noe som reduserte dimensjonaliteten til problemet til alltid å være det minste av antall kovariater eller antall observasjoner. Denne effektiviseringen ga stort utbytte i redusert beregningstid, spesielt når antall kovariater ble vesentlig større enn antall observasjoner.

9.2.4 Valg av datasett

De data som ble benyttet i oppgaven besto i hovedsak av simulerte datasett, i tillegg til at genekspresjonsdata fra ett ekte datasett ble benyttet. Årsaken til at jeg nesten utelukkende benyttet simulerte data er at man ved å simulere data fra en kjent modell kan sammenlikne en modell estimert på bakgrunn av slike data med denne sanne modellen, og si noe om hvor god den estimerte modellen er i forhold til denne.

Overlevelsesdata ble i denne studien simulert fra en Weibullfordeling. Denne fordelingen ble valgt fordi det er en fleksibel og vanlig benyttet fordeling å anta for levetider, samtidig som den gjør det enkelt å inkludere kovariater i en statistisk modell. En av fordelene med å bruke Weibullfordelingen er at den kan representeres både som en akselerert levetidsmodell og som en proporsjonal hasardmodell. Levetidene ble simulert gjennom den akselererte levetidsmo-

dellen, ettersom denne gir en enkel lineær sammenheng mellom kovariater og levetider. Som nevnt i avsnitt 3.8.1 viser Solomon[47] at parameterestimatene for en proporsjonal hasard-modell tilpasset på data som egentlig følger en akselerert levetidsmodell blir komponentvis proporsjonale med parameterne for den sanne underliggende modellen. Dette ble også bekreftet i praksis ved at simulerte data ble tilpasset både med den akselererte levetidsmodellen de var simulert fra, og med Cox-modellen. Detaljene fra resultatene er ikke vist, men begge modellene plukket ut de samme kovariatene som signifikante, til tross for at verdien til de estimerte koeffisientene ble mindre enn de estimerte koeffisientene til den akselererte levetids-modellen. Resultatene viste at det var mulig å estimere en Cox-modell på bakgrunn av denne typen simulerte data.

Metoden for å simulere kovariatdata ble ansett som enkel, effektiv og god nok for å kunne belyse problemstillingene i denne oppgaven. Om ønskelig kunne dataene vært generert med mer sofistikerte simuleringsmetoder. Blant annnet kunne det vært ønskelig å variere uttrykket til gener av betydning i forhold til de uten betydning, eller man kunne gi de ulike genene med betydning ulik effekt, slik at noen gener hadde større effekt på levetiden enn andre. I tillegg ville det være en fordel å kontrollere korrelasjonen mellom genene på en annen måte enn det ble gjort i denne oppgaven. Under simuleringen ble det kun utnyttet at når man trekker p vektorer av endelig lengde n og $p > n$ vil dette medføre at noen av vektorene nødvendigvis vil bli korrelerte. Korrelasjonen kan også skyldes sporadisk korrelasjon, ettersom vi trekker genekspresjonene tilfeldig. En alternativ strategi kunne være selv å påtvinge korrelasjon mellom genekspresjonsvektorene, slik at man har mer kontroll over hvor sterk korrelasjonen er mellom de ulike genene. Det ville ha vært en fordel å teste metodene på flere andre varianter av simulerte datasett, for å se om dette kunne gitt andre resultater enn de vi har sett her. Særlig interessant er det å få de simulerte dataene så like som de reelle dataene som mulig. På grunn av tidsbegrensningen rundt oppgaven ble dette ikke gjort.

Det ekte datsettet ble kun benyttet for å teste effekten av korrelasjon mellom genene i denne studien. Datasettet kunne med fordel ha blitt brukt til å teste flere av metodene i oppgaven, men ettersom man ikke kjenner den ekte modellen for dette datasettet måtte slike tester blitt evaluert etter andre kriterier enn for bruk av de simulerte datasettene. Innenfor denne studien ble det ikke tid til dette, men videre arbeid bør innebære å teste metodene presentert i oppgaven på ekte data og å bestemme evalueringskriterier for tilpasninger på slike datasett.

9.2.5 Effekten av å endre egenskaper ved dataene

Effekten av å endre egenskaper ved dataene ble testet ved å benytte en rekke ulike simulerte datasett. Parameterne ble valgt både slik at det skulle være mulig å si noe om effekten av dem på metodene, og slik at de best mulig representerte ekte datasett. En del av parameterne, som for eksempel antall observasjoner og andelen sensurerte observasjoner sier noe om kvaliteten på datasettene. Det var ønskelig å teste om metodene i denne oppgaven håndterer datasett av varierende kvalitet.

Først ble effekten av å endre antall observasjoner i forhold til antall kovariater testet. Et lite antall gener i forhold til antall observasjoner ga gode resultater for både prediksjon og genseleksjon. Når antall kovariater ble større enn antall observasjoner ble det derimot vanskelig å skille ut gener som har betydning for levetiden fra de som ikke har betydning for levetiden.

Når antall kovariater økte i forhold til antall observasjoner fikk man også dårligere estimater for predikert risiko. Konklusjonen er som man kunne forvente at flere observasjoner gir bedre estimater for modellen. Dette kan sees på som at jo færre observasjoner det er i forhold til kovariater, jo mer ill-posed blir problemet, og det blir vanskeligere å finne gode estimater basert på slike datasett.

Videre ble effekten av å endre antall relevante kovariater i forhold til det totale antallet kovariater testet. Disse resultatene viste at prediksjonen blir dårligere jo større andel av genene som hadde betydning for levetiden. En mulig tolkning av dette tar utgangspunkt i at prediksjonsfeilen $GFPR$ essensielt kan splittes opp i et variansbidrag og biasbidrag for de estimerte parameterne, på samme måte som for MSE (se appendiks A). Man studerer så disse bidragene for hver komponent i dem estimerte parametervektoren $\hat{\boldsymbol{\beta}}(\lambda)$ for seg. Dersom gen i virkelig er uten betydning for levetiden vil $\hat{\beta}_i(\lambda)$ essensielt være normalfordelt med forventning 0 og en varians σ^2. Det eneste bidraget til prediksjonsfeilen fra et slikt estimat vil derfor være variansbidraget. Hvis gen i derimot har en betydning for levetiden vil $\hat{\beta}_i(\lambda)$ bidra både med varians og med bias, ettersom dette er en forventningsskjev estimator. Gener uten betydning bidrar altså bare med varians, mens gener med betydning bidrar med varians i tilnærmet samme størrelse i tillegg til bias. Mange gener uten betydning vil bidra med bare variansledd fra disse estimatene og med det en lavere prediksjonsfeil enn hvis flere gener av betydning blir inkludert. Disse vil bidra både med varians og med bias, noe som øker prediksjonsfeilen. Ut fra resultatene kan det se ut til at prediksjonsfeilen er proporsjonal med andelen gener av betydning inntil denne andelen har blitt 50%, mens når andelen blir større enn dette er det en annen effekt som trer inn og prediksjonsfeilen øker kraftig. Dette kan skyldes at da blir flere av kovariatene med betydning korrelerte, noe som igjen gjør at problemet blir mer ill-posed. Generelt vil korrelerte kovariater får stor varians, og dermed vil også prediksjonsfeilen øke. Om denne tolkningen av resultatene er holdbar kan testes ut ved å gjøre flere simuleringer og se om de samme fenomenene inntreffer. Resultatene viser også, som man kunne forvente, at det er lettere å velge ut gener av betydning jo flere gener det er av betydning. Jo flere gener som har betydning, jo mindre blir sjansen for å gjøre feil dersom man velger ut dette genet som relevant for overlevelsen.

Genekspresjonsdata karakteriseres ofte av at det er høy korrelasjon mellom ekspresjonsvektorene. Effekten av å endre korrelasjonen mellom genene ble derfor testet ut, for å se hvor stor effekt dette hadde på evalueringskriteriene. Resultatene viste at høy korrelasjon mellom genekspresjonsvektorene gjorde det vanskeligere å plukke ut genene av betydning, enn dersom vektorene hadde mindre korrelasjon. Prediksjonen så ut til å være mindre påvirket av korrelasjonen, men lav korrelasjon så ut til å gi bedre prediksjon enn ved høy korrelasjon. Resultatene viste altså at det var lettere å skille ut gener med lav korrelasjon fra de andre genene.

Ekte overlevelsesdata har som regel en viss andel sensurerte individer på grunn av frafall under studien og begrenset oppfølgingstid av pasientene. Det er derfor interessant å studere effekten av andelen sensurerte levetider på hvor lett det er å plukke ut relevante gener og på estimering av predikert risiko. Resultatene viste at ved høy sensurandel ble det problematisk å plukke ut de relevante genene, men for lavere sensurandel enn dette var var det ikke noen markant forskjell i hvordan metodene klarte å plukke ut relevante gener. Derimot får man en bedre prediksjonsfeil jo lavere sensurandelen er. Ut fra dette kan man anta at dersom man har en stor andel sensurerte data i datasettet vil man ikke kunne forvente å få like presise

resultater som når man har en lav sensurandel. Dette skyldes at tilpasningen av Cox-modellen i hovedsak baseres på informasjon fra personer som har opplevd en hendelse, og ved å redusere andelen av disse vil estimering av modellen baseres på mindre data og man vil dermed kunne forvente at dette gir mer usikre estimater.

Alle resultatene presentert for testene av å endre parameterne er basert på kun få estimeringer for hver sammensetning av simuleringsparameterne. Et større antall simuleringer kunne med fordel vært kjørt for å kunne si noe sikkert om effekten av å endre de ulike parameterne, og for å teste enda flere situasjoner enn det ble gjort her. I denne studien ble det ikke tid til å gjennomføre dette.

Evalueringskriterier

Effekten av å endre parameterne ble vurdert etter visse evalueringskriterier for prediksjon av risiko og hvor godt det var mulig å plukke ut gener av betydning. Prediksjon av risiko ble ansett som et godt kriterium, ettersom man ofte er interessert i å kunne predikere overlevelse for nye individer og fordi det ga et godt mål for hvor godt modellen var tilpasset data. Evalueringskriteriene for genseleksjon som ble brukt var svært enkle. Å beregne prosentandelen av de r genene med størst estimert koeffisientverdi som tilsvarte gener med ekte betydning for overlevelsen viste seg å være et greit kriterium, som fungerte i alle situasjoner. Som et alternativ til å plukke ut antall gener på denne måten kunne rangeringen til de estimerte koeffisientverdiene ha vært sammenliknet med rangeringen til den sanne koeffisientvektoren, ved å se på rank-korrelasjon. Dette innebærer å sammenlikne rangeringen av observasjonene i begge koeffisientvektorene. Dette kan gjøres ved å bruke et mål for rank-korrelasjon, for eksempel Spearmans rank-korrelasjon eller en tilsvarende metode. Dersom man får en rangering av de estimerte koeffisientene som tilsvarer den sanne rangeringen vil vi kunne si at vi har en god modelltilpasning, ettersom man da klarer å rangere genene riktig etter hvilken betydning de har for levetiden. Dette alternativet ble det ikke tid til å gjennomføre.

Den alternative metoden til genseleksjon som ble benyttet, ved å lage log-likelihoodplottet for en stegvis utvidet modell, viste seg å være et dårligere kriterium. Bedre metoder for å finne antall gener ut fra et slikt plott kunne med fordel vært benyttet. Ved kun å inspisere plottet ble valg av gener svært subjektivt. I tillegg var det svært beregningskrevende i situasjoner der antall gener ble stort, og dette kriteriet ble derfor ikke benyttet i slike situasjoner.

Det var også ønskelig å teste alternative metoder for hvordan man kan gjøre genseleksjon, og hvorvidt det er mulig å gjøre en genseleksjon på bakgrunn av en straffet Cox-modell i en situasjon som dette. I sammenheng med dette ble metodene pFDR og permutasjonstester presentert. Disse er metoder som gjør det mulig å velge terskelverdier for om et gen skal sies å ha betydning for levetiden eller ikke. Som nevnt i kapittel 7 var det vanskelig å innføre pFDR for den straffede modellen, ettersom det er vanskelig å beregne p-verdier for modellen. Det ble heller ikke tid til å gjennomføre permutasjonstester i denne oppgaven.

9.2.6 Modellseleksjon

Kryssvalidering

Som forslag til metoder for å estimere optimal straffeparameter så vi på tre ulike kryssvalideringsvarianter for Cox-modellen. I kapittel 6 ble det gjort en teoretisk sammenlikning av de tre variantene som viste at de har mye til felles i hvordan de estimerer denne straffeparameteren, men at de har ulike måter å vektlegge risikobidragene til individene i datasettet. Tester av de tre metodene i praksis viste at ved bruk av datasett med forholdsvis mange observasjoner i forhold til antall gener fant alle metodene et estimat for optimal straffeparameter i alle situasjonene som det ble testet for. Det viste seg at to av variantene i disse situasjonene estimerte straffeparameteren λ rimelig likt, mens den siste konsekvent estimerte λ til en høyere verdi enn de to andre. Evalueringer av de estimerte modellene viste at denne siste varianten i disse situasjonene også ga dårligere resultater både for prediksjon av levetider for nye individer og lavere log-likelihoodverdier enn de to andre variantene. En av grunnene til at to av variantene gir liknende resultater kan være likhetene som ble påpekt i avsnitt 6.6, men om dette faktisk er tilfellet er det ikke mulig å gi noe klart svar på ut i fra testene i denne oppgaven.

Videre ble de tre variantene forsøkt testet ut på datasett med mange gener i forhold til antall observasjoner. Dette skapte problemer for estimering av straffeparameteren for de to tidligere publiserte CV-variantene. Det viste seg at uansett hvor stor man valgte straffen λ fantes det ikke noe optimum for kryssvalideringskriteriene for disse metodene. Hva disse resultatene skyldes er vanskelig å si. En mulig forklaring på dette fenomenet er at det oppstår numeriske problemer ved beregning av $\hat{\lambda}$ for disse metodene når datasettet har denne formen. En annen måte å tolke disse resultatene på er å si at disse metodene i situasjoner med mange gener og få observasjoner velger en svært stor straff. Dette vil tilsvare å si at ingen av genene har noe å si for levetiden, eller snarere at data inneholder så mye støy at det ikke er mulig å hente ut eventuell informasjon om levetidene fra dem, til tross for at problemet reguleriseres. Den beste modellen i slike tilfeller vil derfor bli modellen der ingen informasjon om kovariatene inkluderes. Dette er isåfall ikke en feil med disse metodene, men reflekterer snarere en konservativ holdning til hvorvidt kovariater bør inkluderes i modellen. På den andre side er ikke dette særlig konstruktivt, dersom man ønsker å estimere en modell for å kunne predikere levetider basert på genekspresjonsdata eller å plukke ut gener med relevans for levetiden.

Min foreslåtte variant ga derimot et estimat for optimal straffeparameter for alle valg av parametere det ble testet for. De estimerte modellene ble sammenliknet med den optimale straffede estimerte modellen funnet ved å bruke orakelestimatoren ved å bruke feilmålet Err som beregner log_2-ratioen av prediksjonsfeilen til CV-variant 3 og orakelestimatoren. Resultatene viste som tidligere sett at flere observasjoner i forhold til antall kovariater ga bedre prediksjonsestimater for de estimerte modellene. Err ble mindre jo flere observasjoner treningssettet inneholdt. Videre ble Err for treningssettet mindre enn for testsettene, noe som var forventet ettersom modellen er tilpasset ved å bruke treningssettet og vil med det som regel gjøre en bedre prediksjon for dette enn for uavhengige testdata. Resultatene viser altså at det er mulig å finne en optimal estimert modell ved å bruke min foreslåtte variant, selv i situasjoner der antall kovariater er stort i forhold til antall observasjoner. Ut fra feilmålet brukt i denne oppgaven er det derimot vanskelig å si hvor god en slik estimert modell vil være

til å predikere overlevelse eller risiko i praksis. Feilmålet Err sier bare hvor godt den estimerte modellen predikerer i gjennomsnitt i forhold til orakelestimatoren, ikke noe om hvor godt man kan predikere overlevelse for nye individer. For å kunne si noe om hvor god modellen faktisk er, for eksempel om den er brukbar til for eksempel å predikere overlevelse i praksis, kunne man sett på andre feilmål. Et mulig forslag ville være å se på hvor stor feil i prediksjon som gjøres per individ og finne et estimat for variansen til prediksjonsfeilen. Et slikt estimat vil kunne si noe om hvor stor varians man kan forvente i en predikert risiko for et nytt individ som inkluderes.

Hva som gjør at mitt forslag til kryssvalidering for Cox-modellen faktisk finner en optimal straffeparameter, men ikke de to andre har det i denne studien ikke vært mulig å avdekke, men det viser at denne nye CV-varianten kan ha gode egenskaper for bruk i denne typen situasjoner. I tillegg viste resultatene at denne metoden ga bedre estimater for prediksjon av overlevelse enn begge de to andre variantene når de ble testet på datasett med færre kovariater og mange observasjoner. Dette er med på å støtte opp under at dette er et modellseleksjonskriterium for Cox-modellen som det vil være interessant å se nærmere på.

Resultatene i denne oppgaven viser at det er helt klart potensiale for videre studier av disse kryssvalideringsmetodene. De tidligere publiserte metodene fungerer ikke i alle situasjoner, og er derfor ikke en universal løsning på hvordan man bør gjøre kryssvalidering for en Cox-modell. Det er heller ikke noe som tilsier, fra de teoretiske eller praktiske betraktningene av metodene, at den ene skal være bedre enn noen av de andre. Videre studier av disse vil kunne avdekke egenskaper ved metodene som kan hjelpe oss i valget av kryssvalideringsmetode i situasjoner som vi har sett på i denne oppgaven. Det har vært gjort omfattende litteratursøk for å finne andre studier som beskriver liknende sammenlikninger av kryssvalideringsvarianter for Cox-modellen, uten at noe ble funnet. Det er altså nødvendig å teste dem ut på flere datasett før man kan si noe generelt om hvordan de presterer, eventuelt hvilke av dem som er best. Litteratursøket resulterte heller ikke i at det ble funnet studier som beskriver tester av de allerede publiserte metodene i bruk på datasett med så mange kovariater som det er snakk om her. Testene utført i denne oppgaven gir en ide om hvordan metodene oppfører seg i slike situasjoner, og tilsier at det vil være interessant å studere alle de tre variantene videre. Det ville også vært interessant å studere flere kriterier å vurdere de ulike variantene etter, for eksempel å kunne utføre en form for hypotesetesting for den estimerte parametervektoren for Cox-modellen.

L-kurve kriteriet

En siste form for modellseleksjon som ble testet var L-kurve kriteriet. Denne har vært i bruk som modellseleksjonsmetode innen andre fagfelt enn statistikk, men kriteriet har såvidt meg bekjent ikke vært benyttet i sammenheng med en straffet log-likelihood (et litteratursøk ga heller ingen resultater).

Testinger av L-kurvekriteriet ga uventede resultater. Kurvene fikk i mange tilfeller en annen form enn det som var forventet. Ettersom kurvene får en horisontal form for små verdier av λ kan dette tyde på at det er log-likelihooden som er mest følsom for at $\boldsymbol{\beta}$ straffes litt, mens den estimerte $\boldsymbol{\beta}$ ikke endres så mye før straffen øker til større verdier. Dette kommer fram i at kurven får en nær vertikal form når straffen blir større.

Det ble i første omgang gjort et forsøk på å tolke dette. En egenskap ved ridge-type straff er at ved lav straff straffes de små komponentene av $\boldsymbol{\beta}$ mest, det vil si de som er nær 0. Dette vil ikke ha så stor effekt på normen til $\boldsymbol{\beta}$ ettersom antallet gener uten betydning var stort, og verdien disse koeffisientene var små i utgangspunktet. Derimot har det stor effekt på log-likelihooden ettersom denne er svært følsom for om komponentene i $\boldsymbol{\beta}$ er 0 eller ikke. Når straffen øker begynner også de komponentene av $\boldsymbol{\beta}$ som er mer ulik 0 å reguleres. Dette vil ha en mindre drastisk effekt på log-likelihooden, men det vil få større effekt på normen til $\boldsymbol{\beta}$. Dette kunne vært en mulig forklaring på den uventede kurveformen. Videre tester viste derimot at dette ikke så ut til å stemme. Dersom antall kovariater av betydning var stort ville man forventer å se en annen form på kurven motsatte, men resultatene viste at dette ikke var tilfellet. Til tross for videre litteraturstudier og tester var det ikke mulig innenfor denne oppgavens tidsramme å finne noen god forklaring på hvorfor kurvene får denne formen.

9.3 Konklusjon og videre arbeid

Resultatene fra denne oppgaven viser at det er mulig å tilpasse en Cox-modell med ridge-type straff på bakgrunn av genekspresjonsdata fra mikroarrayer ved å bruke de metodene som har blitt presentert i denne oppgaven. Resultatene viste at datakvaliteten har stor innvirkning på hvor gode estimater vi får, men det er vanskelig å si noe om hvor gode metodene er i praksis ettersom de ikke har blitt testet ut på ekte datasett. Evalueringskriteriene som ble brukt ga stort sett muligheten til å sammenlikne ulike situasjoner, men sier lite om hvor gode metodene er for ekte datasett, for eksempel hvor godt det er mulig å predikere risikoen for nye individer.

Vi har sett at tilpasning og estimering av en straffet modell på bakgrunn av genekspresjonsdata byr på en del utfordringer. Både modellseleksjon og genseleksjon er utvilsomt vanskelig. Vi har blant annet sett at tidligere publiserte løsninger for modellseleksjon ikke nødvendigvis er patente løsninger, og at det trengs videre arbeid på dette området.

Problemstillinger der man har mange kovariater og få observasjoner har en sentral plass i bioinformatikk såvel som en del andre felter. Vi kan velge å betrakte dette som et problem som bør løses ved å prøve å omforme hele problemet tilbake til en kjent situasjon, ved å kutte ned på antall variable. Alternativt kan vi velge å snu hele problemet på hodet og si at den beste måten å håndtere slike problemer på er å er å utvikle nye metoder som takler denne typen problemstillinger. Man kan se det som positivt å ha med mange kovariater, ettersom man ved å ha mange kovariater faktisk har mer informasjon om hvert individ enn med få. Istedenfor å tenke at de ulike kovariatene overskygger hverandre med støy kan man heller se på det som at informasjonen i de ulike kovariatene kan bekrefte hverandre, og jo flere kovariater man har med, jo mer informasjon får man også med.

Et forslag til mulig videre arbeid vil være å forfølge den kryssvalideringsvarianten som ble presentert i denne oppgaven, et alternativ som i følge resultatene i denne oppgaven ser lovende ut. Videre arbeid med L-kurve kriteriet vil også være interessant. Dette vil innebære å prøve å forstå hvorfor kurvene får den formen de gjør. En mulig framgangsmåte vil være å se om det har vært gjort tidligere studier av L-kurvene i situasjoner med mange kovariater og få observasjoner. Erfaringer fra dette kan muligens overføres til den situasjonen som har blitt studert her. En videre analyse av kurvene kunne også avdekket om det maksimale krumnings-

punktet på disse kurvene har en nyttig tolkning i denne sammenhengen. En dypere forståelse av dette fenomenet vil kunne føre til at L-kurvekriteriet vil kunne vurderes opp mot de andre modellseleksjonsmetodene.

Videre arbeid vil også kunne inkludere å teste ut alle metodene på ekte datasett og på andre varianter av simulerte datasett, slik det ble beskrevet i diskusjonen over. Blant annet kunne det vært interessant å sammenlikne resultatene fra metodene som har blitt presentert i denne oppgaven med resultater fra andre verktøy for analyse av genekspresjonsdata, som for eksempel SAM. Man burde i tillegg jobbe videre med bedre metoder for å gjøre genseleksjon for en straffet Cox-modell, spesielt kunne det vært interessant å jobbe videre med bruk av pFDR for å plukke ut interessante gener. For å gjøre dette må man kunne regne ut p-verdier for estimatene i den straffede modellen. En tilnærming til å se på beregning av p-verdier for en straffet Cox-modell vil være å starte med å studere standard ridge regresjon for en ordinær lineær modell der man har en fast straff λ, og finne et uttrykk for variansen til parameterestimatene i slike tilfeller. Deretter kan man gå videre og se om det er mulig å gjøre det samme i situasjoner der straffeparameteren ikke lenger er fast, men må estimeres. Videre kan man om mulig prøve å overføre erfaringer fra disse studiene til den mer kompliserte situasjonen der vi har en straffet Cox-modell, igjen i tilfeller der straffeparameteren er fast og når den må estimeres.

Bibliografi

[1] Odd O. Aalen, Tron Anders Moger, Ellen J. Amundsen, Solve Sæbø og Harald Weedon-Fekjær. Overlevelses- og forløpsanalyse: Det tematiske området norevent. *Norsk Epidemiologi*, 13(2):233–238, 2003.

[2] Dhammika Amaratunga og Javier Cabrera. *Exploration and analysis of DNA microarray and protein array data*. Wiley series in probability and statistics. Wiley, 2004.

[3] Per Kragh Andersen og Michael Væth. *Statistisk analyse af overlevelsesdata*. FADLs forlag, 1984.

[4] Eric Bair og Robert Tibshirani. Semi-supervised methods to predict patient survival from gene expression data. *PLOS Biology*, 2(4):511–522, 2004.

[5] David G. Beer, Sharon L. R. Kardia, Chiang-Ching Huang, Thomas J. Giordano, Albert M. Levin, David E. Misek, Lin Lin, Guoan Chen, Tarek G. Gharib, Dafydd G. Thomas, Michelle L. Lizyness, Rork Kuick, Saturo Hayasaka, Jeremy M. G. Taylor, Iannettoni Mark D., Mark B. Orringer og Samir Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature Medicine*, 8(8):816–824, 2002.

[6] Yoav Benjamini og Yosef Hochberg. Controlling the false discovery rate: A practical and poweful approach to multiple testing. *Joural of the Royal Statistical Society. Series B*, 57(1):289–300, 1995.

[7] N. Breslow. Covariance analysis of censored survival data. *Biometrics*, 30:89–99, 1974.

[8] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society B*, 34:187–202, 1972.

[9] D. R Cox og D. Oakes. *Analysis of survival data*. Chapman and Hall, 1984.

[10] S. De Jong. Pls shrinks. *Journal of Chemometrics*, (9):323–326, 1995.

[11] J. E. Dennis Jr. og Robert B. Schnabel. *Numerical methods for unconstrained Optimization and Nonlinear equations*. Society for industrial and applied mathematics, 1983.

[12] Richard O. Duda, Peter E. Hart og David G. Stork. *Pattern Classification*. John Wiley & Sons,Inc., 2 utgave, 2000.

[13] Sandrine Dudoit, Jane Fridlyand og Terence P. Speed. Comparison of discrimination methods for the classification of tumors usine gene expression data. *Teknisk rapport no. 576, UC Berkeley*, juni 2000.

[14] Jianbing Fan, Diping Che, Chanfeng Zhao, Lixin Zhou og Weinyi Feng. High-density fiber optic arrays technology and its applications in functional genomic studies. *Chinese science bulletin*, 48(18):1903–1905, 2003.

[15] Constantinos Goutis. Partial least squares algorithm yields shrinkage estimators. *The Annals of Statistics*, 24(2):816–824, 1996.

[16] Chares W. Groetsch. *Inverse problems in the mathematical sciences*. Vieweg, 1993.

[17] Jacques Hadamard. *Lectures on Cauchy's problem in linear partial differential equations*. Dover publications, 1952.

[18] Per Christian Hansen. *Rank-deficient and discrete ill-posed problems*. SIAM, 1997.

[19] Trevor Hastie og Robert Tibshirani. Expression arrays and the p≫n problem. *Not known*, 2003.

[20] Trevor Hastie, Robert Tibshirani, Michael B. Eisen, Ash Alizadeh, Ronald Levy, Louis Staudt, Wing C. Chan, David Botstein og Patrick Brown. 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome biology*, 1(2), 2000.

[21] Trevor Hastie, Robert Tibshirani og Jerome Friedman. *The elements of statistical learning*. Springer series in statistics. Springer, 2001.

[22] Arthur E. Hoerl og Robert W. Kennard. Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12:69–82, 1970.

[23] Arthur E. Hoerl og Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), feb 1970.

[24] Marit Holden og Anders Løland. Introduksjon til analyse av cdna mikromatrisedata. *Norsk epidemiologi*, 13(2):291–296, 2003.

[25] Philip Hougaard. *Analysis of Multivariate Survival Data*. Statistics for Biology and Health. Springer, 2000.

[26] Hans C. van Houwelingen. Shrinkage and penalized likelihood as methods to improve prediction accuracy. *Statistica Neerlandica*, 55(1):17–34, 2001.

[27] T. C. Hsiang. A bayesian view on ridge regression. *The Statistician*, (4):267–268, dec 1975.

[28] John P. Klein og Melvin L. Moeschberger. *Survival analysis - Techniques for Censored and Truncated Data*. Statistics for Biology and Health. Springer, 1997.

[29] Anthony Y. C. Kuk. All subsets regression in a proportional hazards model. *Biometrika*, 71(3):587–592, Des 1984.

[30] Steven J. Leon. *Linear Algebra with Applications*. Prentice Hall, 6 utgave, 2002.

[31] Hongzhe Li og Yihui Luan. Kernel cox regression models for linking gene expression profiles to censored survival data. *Vet ikke dette*, 00:00, 0000.

[32] D. V Lindley og A. F. M. Smith. Bayes estimates for the linear model. *Journal of the Royal Statistical Society B*, 34:1–19, 1972.

[33] Ole Christian Lingjærde. An introduction to numerical methods for unconstrained optimization. Teknisk rapport, 1998.

[34] Ole Christian Lingjærde og Nils Christoffersen. Shrinkage structure of partial least squares. *Scandinavian journal of statistics*, 27:459–473, 2000.

[35] D. Lockhart, H. Dong, M. Byrne, M. Follettie, M. Gallo, M. Chee, M. Mittmann, C. Wang, M. Kobayaski, H. Horton og E. Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature biotehnology*, 14:1675–1680, 1996.

[36] Izidore S. Lossos, Debra K. Czerwinski, Ash Alizadeh, Mark A. Wechser, Rob Tibshirani, David Botstein og Ronald Levy. Prediction of survival in diffuse large-b-cell lymphoma based on the expression of six genes. *The New England journal of medicine*, 350(18):1828–1837, 2004.

[37] S. A. Murphy og A. W. van der Vaart. On profile likelihood. *Journal of the American Statistical Association*, 95(450), 2000.

[38] Danh V. Nguyen og David M. Rocke. Partial least squares proportional hazard regression for application to dna microarray survival data. *Bioinformatics*, 18(12):1625–1632, 2002.

[39] Sean Fionnbarr O'Suilleabhain. *The analysis of some penalized likelihood estimation schemes*. Doktorgradsoppgave, The University of Wisconsin-Madison, 1983.

[40] Peter J. Park, Lu Tian og Kohane Isaac S. Linking gene expression data with patient survival times using partial least squares. *Bioinformatics*, 18:120–127, 2002.

[41] Charles M. Perou, Therese Sørlie, Michael B. Eisen, Matt van de Rijn, Stefanie S. Jeffrey, Christian A. Rees, Jonathan R. Pollack, Douglas T. Ross, Hilde Johnsen, Lars A. Akslen, Øystein Fluge, Alexander Pergamenschikov, Cheryl Williams, Shirley X. Zhu, Per E. Lønning, Anne-Lise Børresen-Dale, Patrick O. Brown og David Botstein. Molecular portraits of human breast tumors. *Nature*, 456, 2000.

[42] William H. Press, Saul A. Teukolsky, William T. Vetterling og Brian P. Flannery. *Numerical Recipes in C*. Cambridge university press, 2 utgave, 1992.

[43] John Quackenbush. Microarray data normalization and transformation. *Nature Genetics Supplement*, 32, dec 2002.

[44] Brian D. Ripley. *Stochastic simulation*. John Wiley & sons, 1987.

[45] Andreas Rosenwald, George Wright, Wing C. Chan, Joseph M. Connors, Elias Campo, Richard I. Fisher, Randy D. Gascoyne, Konrad Muller-Hermelink, Erlend B. Smeland og Louis M. Staudt. The use of molecular profiling to predict survival after chemotherapy for diffuse large-b-cell lymphoma. *The New England Journal of Medicine*, 346(25), 2002.

[46] Nils Olav Sjøberg. *Molekylær genetikk - Genteknologi - humant DNA*. Vett og viten, 3 utgave, 2002.

[47] Patricia J. Solomon. Effect of misspecification of regression models in the analysis of survival data. *Biometrika*, 71(2):291–298, aug 1984.

[48] John D. Storey og Robert Tibshirani. Statistical significance for genomewide studies. *PNAS*, 100(16):9440–9445, 2003.

[49] Peter Sudbery. *Human molecular genetics*. Prentice Hall, 2 utgave, 2002.

[50] Therese Sørlie, Charles M. Perou, Robert Tibshirani, Turid Aas, Stephanie Geisler, Hilde Johnsen, Trevor Hastie, Michael B. Eisen, Matt van de Rijn, Stefanie S. Jeffrey, Thor Thorsen, Hanne Quist, John C. Matese, Patrick O. Brown, David Botstein, Per Eystein Lønning og Anne-Lise Børresen-Dale. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *PNAS*, 98(19):10869–10874, september 2001.

[51] Robert Tibshirani. The lasso method for variable selection in the cox model. *Statistics in medicine*, 16:385–395, 1997.

[52] Pierre J. M. Verweij og Hans C. van Houwelingen. Cross-validation in survival analysis. *Statistics in medicine*, 12:2305–2314, 1993.

[53] David R. Walt. Bead-based fiber-optic arrays. *Science*, 287:451–452, 2000.

[54] M. A Wolfe. *Numerical methods for unconstrained optimization*. Van Nostrand Reinhold company, 1978.

[55] Joanne M. Yeakley, Jian-Bing Fan, Dennis Doucet, Lin Lou, Eliza Wickham, Zhen Ye, Mark S. Chee og Xiang-Dong Fu. Profiling alternative splicing on fiber-optic arrays. *Nature biotechnology*, 20:353–358, 2002.

[56] Hui Zou og Trevor Hastie. Regression shrinkage and selection via the elastic net, with applications to microarrays. *Teknisk rapport*, 2003.

Tillegg A

Parameterestimering

Vi ønsker ofte å tilpasse modeller på bakgrunn av en mengde observerte data på formen $D = ((y_1, \mathbf{x}_1, \ldots, (y_n, \mathbf{x}_n))$. Dersom vi har en parametrisk modell gjøres dette ved å estimere en parametervektor $\boldsymbol{\beta}$. I en regresjonssammenheng vil $\boldsymbol{\beta}$ kunne være koeffisientvektoren vi ønsker å estimere for å kunne si noe om sammenhengen mellom en mengde forklarinsvariable og en responsvariabel. Det er to vanlige måter å estimere en slik parametervektor på, nemlig med bayesiansk estimering eller med en frekventistisk tilnærming kalt maksimum likelihoodestimering. Begge tilnærminger er mye brukt. De resulterer ofte i tilnærmet samme resultat, selv om tilnærmingene er konseptuelt ulike.

A.1 Frekventistisk tilnærming til estimering

Frekventist-tilnærmingen til statistikk anser parametere som kvantiteter med faste, men ukjente verdier og dataene som realiseringer av stokastiske prosesser. Den vanligste metoden å estimere parametere på kalles maksimum likelihood estimering (ML). Estimatet for $\boldsymbol{\beta}$ er den verdien som maksimerer sannsynligheten for de observasjonene vi har registrert. Dersom vi har en sannsynlighetstetthet $f(\mathbf{x}|\boldsymbol{\beta})$ ønsker vi å estimere parametervektoren $\hat{\boldsymbol{\beta}}$ ut fra denne. Anta at vi har en mengde treningsobservasjoner $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ som er uavhengig trukket fra denne fordelingen. Da er

$$L(D, \boldsymbol{\beta}) = p(D|\boldsymbol{\beta}) = \prod_{i=1}^{n} f(\mathbf{x}_i|\boldsymbol{\beta})$$

Dersom vi ser på L som en funksjon av $\boldsymbol{\beta}$ der D holdes fast kalles dette for *likelihoodfunksjonen* eller bare *likelihooden* til $\boldsymbol{\beta}$ med hensyn på observasjonsmengden, og skriver da $L(\boldsymbol{\beta})$. Dersom $L(D, \boldsymbol{\beta})$ blir sett på som en funksjon av D med $\boldsymbol{\beta}$ fast er den isteden en sannsynlighetstetthetsfunksjon. En likelihood er ikke det samme som en sannsynlighet, for eksempel så summerer ikke integralet under er likelihoodfunksjon til 1.

Et maksimumlikelihood-estimat $\hat{\boldsymbol{\beta}}$ er en verdi av $\boldsymbol{\beta}$ som maksimerer denne funksjonen. Det kan sees på som en verdi som best stemmer med eller støtter de observasjonene vi har gjort.

Analytisk sett kan det ofte være lettere å jobbe med logaritmen av likelihooden, *log-likelihooden*, $l(\boldsymbol{\beta})$ enn med selve likelihooden. Ettersom logaritmen er monotont økende vil den $\hat{\boldsymbol{\beta}}$ som maksimerer likelihooden også maksimere log-likelihooden. Dersom $l(\boldsymbol{\beta})$ er en veldefinert deriverbar funksjon kan $\hat{\boldsymbol{\beta}}$ finnes ved standard kalkulus ved å maksimere

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} ln\ f(\mathbf{x}_i|\boldsymbol{\beta}),$$

med hensyn på $\boldsymbol{\beta}$. De partiellderiverte er gitt ved

$$\frac{\partial l}{\partial \beta_q} = \sum_{i=1}^{n} \frac{\partial l}{\partial \beta_q}\ l(\boldsymbol{\beta}),$$

og løsningen $\hat{\boldsymbol{\beta}}$ på optimeingsproblemet tilfredsstiller de p likningene $\frac{\partial l}{\partial \beta_q} = \mathbf{0}$ for $q = 1, 2, \ldots, p$.

Estimatet $\hat{\boldsymbol{\beta}}$ kan da være et globalt maksimum, et lokalt maksimum, et sadelpunkt eller tilsvarende minimum. Dersom vi finner alle løsninger er vi garantert å finne den sanne løsningen $\boldsymbol{\beta}$. Ellers må vi stole på informasjon om den annenderiverte for å sikre oss at vi har et maksimum. Dersom $l(\boldsymbol{\beta})$ er konkav må $\hat{\boldsymbol{\beta}}$ være et globalt minimumspunkt, og hvis $l(\boldsymbol{\beta})$ i tillegg er strengt konkav er $\hat{\boldsymbol{\beta}}$ entydig gitt. Maksimum likelihoodestimatene har gode konvergensegenskaper. Det vil si at når antall observasjoner blir stor kan vi forvente å estimere den sanne verdien $\boldsymbol{\beta}$. Et standard resultat vier at fordelingen til ML-estimatoren har en asymptotisk normalfordeling $\hat{\boldsymbol{\beta}} \rightarrow N(\boldsymbol{\beta}, i(\boldsymbol{\beta})^{-1})$ når $n \rightarrow \infty$, der $i(\boldsymbol{\beta})$ er forventningen til informasjonsmatrisen, dvs. $i(\boldsymbol{\beta}) = E[I(\boldsymbol{\beta})]$ der $E[I(\boldsymbol{\beta})_{i,j}] = -E\left[\frac{\partial^2 l(\boldsymbol{\beta}, X)}{\partial \boldsymbol{\beta}_i \partial \boldsymbol{\beta}_j}\right]$.

A.2 Bayesiansk estimering

Bayesiansk statistikk anser parameteren som en stokastisk variabel med en kjent *priorfordeling* og dataene som faste. På grunnlag av dataobservasjoner kan vi modifisere parameterfordelingen, eller om vi vil overføre priorfordelingen til en *a posteriori fordeling*. Dette endrer vårt syn på de sanne verdiene til parameterne. Vi ønsker altså å estimere denne a posteriori fordelingen ved å bruke all tiljengelig informasjon. Denne informasjonen innebærer all forkunnskap vi har om parameterne og informasjon som ligger i de observerte dataene. Vi må anta en priorfordeling for parameteren. Denne kan være basert på hva som helst, gjerne nyttig forkunnskap vi har om fordelingen til parameteren. Vi bruker så en mengde D av observasjoner trukket fra den faste men ukjente sannsynlighetsfordelingen $p(D)$ til å bestemme $p(\boldsymbol{\beta}|D)$. Denne finner vi så ved å bruke Bayes formel

$$p(\boldsymbol{\beta}|D) = \frac{p(D|\boldsymbol{\beta})\ p(\boldsymbol{\beta})}{p(D)}$$

Moden til denne fordelingen er da parameterestimatet.

Dersom vi har fornuftige priorfordelinger og uendelig med data vil resultatet av ML og bayesians estimering i mange tilfeller gi nært sammenfallende resultater. ML har den fordelen at den ofte gir en mindre kompleks utregning enn bayesiansk estimering. Bayesianske metoder

utnytter derimot ofte informasjonen vi har i data bedre, og gir mer eksplisitte uttrykk for problemer rundt bias og varians til estimatorene. Det er gode teoretiske og metodiske argumenter for å begge tilnærminger, men i praksis er ML-estimering enklere å utføre og gir ofte tilnærmet like nøyaktige svar.

A.3 Valg mellom parametere

Et generelt mål for hvor god en parameter er det som kalles gjennomsnittlig kvadrert feil (MSE), som tar høyde for både varians og bias til en parameter. For en generell estimator $\hat{\boldsymbol{\beta}}$ av en parameter $\boldsymbol{\beta}$ er MSE($\boldsymbol{\beta}$) definert som

$$MSE(\boldsymbol{\beta}) = E[||\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}||^2],$$

hvor $||\cdot||$ angir kvadratisk norm (L_2-norm). MSE($\hat{\boldsymbol{\beta}}$) kan uttrykkes som en sum av varians og bias for $\hat{\boldsymbol{\beta}}$ ved å skrive

$$\begin{aligned} MSE(\hat{\boldsymbol{\beta}}) &= E[||(\hat{\boldsymbol{\beta}} - E(\hat{\boldsymbol{\beta}})) + (E(\hat{\boldsymbol{\beta}}) - \boldsymbol{\beta})||^2] \\ &= E[||\hat{\boldsymbol{\beta}} - E(\hat{\boldsymbol{\beta}})||^2] + ||E(\hat{\boldsymbol{\beta}}) - \boldsymbol{\beta}||^2 + 2(E(\hat{\boldsymbol{\beta}}) - \boldsymbol{\beta})^T E[\hat{\boldsymbol{\beta}} - E(\hat{\boldsymbol{\beta}})] \end{aligned}$$

Legg merke til at $E[\hat{\boldsymbol{\beta}} - E(\hat{\boldsymbol{\beta}})] \equiv \mathbf{0}$, at første termen er variansen til $\hat{\boldsymbol{\beta}}$ og den andre termen er den kvadrerte normen til biasen. Vi får derfor at

$$MSE(\hat{\boldsymbol{\beta}}) = var(\hat{\boldsymbol{\beta}}) + ||bias(\hat{\boldsymbol{\beta}})||^2$$

Dersom vi må velge mellom to forventningsskjeve estimatorer velger vi den med minst MSE. Selv om MSE kan virke som en fornuftig måte å bestemme kvaliteten til en estimator, så vil den ikke gi et brukbart kriterium for valg av parameter. Vi ser av uttrykket for MSE at det avhenger av at vi kjenner den sanne parameterverdien $\boldsymbol{\beta}$, og dersom vi hadde gjort det hadde vi allerede vært i mål, ettersom den sanne verdien ville være den som minimerer MSE. Det er derfor ikke praktisk mulig å bruke MSE som kriterium for å velge estimator, og vi trenger derfor et annet kriterium som leder til en praktisk brukbar metode for å velge estimator.

Et alternativ er på bruke prediksjonsfeil som et mål på hvor god parameteren er. Vi kan da bruke en loss-funksjon $L(Y, f(X)$ som straffer prediksjonsfeil. En av de vanligste loss-funksjonene er den kvadratiske loss-funksjonen $L(Y, f(X)) = (Y - f(X))^2$. Treningsfeil blir da gjennomsnittet av loss over treningssettet $\frac{1}{N}\sum_{i=1}^{n} L(y_i, \hat{f}(\mathbf{x}_i)$. Dette er et estimat for hvor godt modellen kan predikere nye observasjoner. Ettersom vi beregner dette fra treningssettet som vi også brukte til å estimere modellen vil dette bli et optimistisk estimat for prediksjon.