Abstract

In this master's thesis the problem of simulating conditional Bernoulli distributed stochastic variables, given the sum, is considered. Three simulation methods are considered, namely the acceptance/rejection technique, Bondesson's method and the Markov chain Monte Carlo method.

To compare the three methods the bias and the standard deviations of the simulated variables are evaluated. The results of the simulation study shows that the Markov chain Monte Carlo method is not the best method for this type of simulation. Both the other methods were quite suitable for the task. The acceptance/rejection technique is a little bit more time consuming than Bondesson's method, but on the other hand the acceptance/rejection technique is easier to implement.

Innehållsförteckning

Jämförelse av olika metoder att generera Bernoullifördelade slumptal givet deras summa

av

Anders Öhlund

Examensarbete i matematisk statistik
Umeå Universitet, 2000

Handledare: Leif Nilsson

1 Inledning

Ibland kan det inom t.ex matematisk statistik vara mycket tidskrävande att lösa ett problem exakt. Då kan ett alternativ vara att lösa problemet med simulering. Det innebär att man inte får den exakta lösningen men man kan ofta få en ganska bra approximation av lösningen inom en rimlig tid.

Simulering har tillämpats inom ämnet matematisk statistik sedan början av 1900-talet men har först på senare år kommit till sin rätt i och med den snabba utvecklingen av datorer. I dag är simulering en viktig del av detta ämne.

I detta examensarbete betraktas problemet med att generera betingade vektorer $\mathbf{X}=(X_1,\ldots,X_k)$ givet $\Sigma_{i=1}^{k} X_i = s$, där X_i är oberoende bernoullifördelade stokastiska variabler och *s* är ett tal mellan *0* och *k*. Bestämning av t.ex. väntevärden givet $\Sigma_{i=1}^{k} X_i = s$ kan vara tidskrävande och simulering kan i sådana fall vara ett alternativ. I denna rapport jämförs tre simuleringsmetoder som benämns Förkastelsemetoden, Bondessons metod samt McMC-metoden.

I Förkastelsemetoden simuleras vektorer från den obetingade fördelningen. Av dessa accepteras de vektorer som summerar till *s*. Förkastelsemetoden blir i många fall långsam på grund av att många vektorer förkastas. I denna rapport kommer vi att använda en metod, föreslagen av Broström och Nilsson (1999)[1], där man genom att transformera sannolikheterna p_i på ett lämpligt sätt kan göra Förkastelsemetoden snabbare samtidigt som man behåller den ursprungliga betingade sannolikhetsfunktionen. Denna metod finns beskriven i kapitel 2.

Med hjälp av definitionen på betingad sannolikhet kan den sökta betingade fördelningen skrivas som en produkt av *k* stycken endimensionella betingade fördelningar. Genom att rekursivt simulera från dessa endimensionella betingade fördelningar erhålls en simulerad vektor från den sökta betingade fördelningen. Denna simuleringsmetod kommer här att gå under namnet Bondessons metod.

Ytterligare en metod i denna rapport är Markov chain Monte Carlo (McMC) metoden. I McMC använder man sig av markovkedjor för att skapa en följd av beroende vektorer från den betingade fördelningen. I rapporten beskrivs Metropolis-Hastings algoritm som är en McMC-metod. Vidare beskrivs också Gibbs sampler som är ett specialfall av Metropolis-Hastings algoritm. Gibbs sampler är den McMCmetod som vi kommer att betrakta.

I kapitel 2, 3 och 4 beskrivs de tre simuleringsmetoderna. Här finns också några exempel på hur de fungerar i praktiken. I kapitel 5 beskrivs en simuleringsstudie som används på de tre simuleringsmetoderna. I kapitel 6 redovisas resultat från simuleringsstudien där de tre metoderna jämförs. Dessa resultat finns redovisade i bilaga 1. Vi ser att för denna simulering passar Förkastelsemetoden och Bondessons metod ungefär lika bra. Man kan vidare se att McMC passar lite sämre än de övriga metoderna i detta fall.

2 Förkastelsemetoden

När man simulerar med Förkastelsemetoden genererar man först vektorer $\boldsymbol{x}=(x_1,x_2,\ldots,x_k)$ från den obetingade stokastiska vektorn $\boldsymbol{X}=(X_1,X_2,\ldots,X_k)$, där X_i är 1 med sannolikheten p_i och 0 med sannolikheten $1-p_i$, dvs Bernoulli(p_i). De vektorer som sedan uppfyller betinget, i detta fall $\Sigma_{i=1}^{k} X_i = s$, accepteras och övriga förkastas. Den stora nackdelen med att simulera med Förkastelsemetoden rakt av är att det är stor risk att vektorer förkastas och metoden blir förhållandevis långsam. Detta gäller speciellt om $E[\Sigma_{i=1}^{k} X_i] = \Sigma_{i=1}^{k} p_i$ skiljer sig mycket från s.

För att Förkastelsemetoden skall fungera tillfredsställande måste man hitta ett sätt att få fler vektorer accepterade. Det man kan göra för att förbättra metoden är att transformera om sannolikheterna p_i på ett sådant sätt att man får fler accepterade vektorer samtidigt som man behåller samma betingade sannolikhetsfunktion som man hade före transformationen. En transformation som uppfyller detta är

$$P_i(\theta) = \frac{p_i\theta}{p_i\theta + 1 - p_i} \qquad , \theta\in(0,\infty).$$

Detta kan inses genom att låta $\boldsymbol{Y(\theta)}=(Y_1(\theta),Y_2(\theta),\ldots,Y_k(\theta))$ vara en vektor med *Bernoulli*$(P_i(\theta))$ stokastiska variabler, $\theta\in(0,\infty)$. Då kan den betingade sannolikhetsfunktionen för $\boldsymbol{Y(\theta)}$ givet $\Sigma_{i=1}^{k} Y_i(\theta) = s$ skrivas som

$$P(\boldsymbol{Y}(\boldsymbol{\theta}) = \boldsymbol{y} \Big| \sum_{i=1}^{k} Y_i(\theta) = s) = \frac{\prod_{i=1}^{k} \left(\frac{p_i\theta}{1+p_i(\theta-1)}\right)^{z_i} \left(\frac{1-p_i}{1+p_i(\theta-1)}\right)^{1-z_i}}{\sum_{\boldsymbol{z}} \prod_{i=1}^{k} \left(\frac{p_i\theta}{1+p_i(\theta-1)}\right)^{z_i} \left(\frac{1-p_i}{1+p_i(\theta-1)}\right)^{1-z_i}} =$$

$$= \frac{\prod_{i=1}^{k} \frac{p_i^{z_i}(1-p_i)^{1-z_i}\theta^{z_i}}{1+p_i(\theta-1)}}{\sum_{\boldsymbol{z}} \prod_{i=1}^{k} \frac{p_i^{z_i}(1-p_i)^{1-z_i}\theta^{z_i}}{1+p_i(\theta-1)}} = \frac{\theta^s \prod_{i=1}^{k} p_i^{z_i}(1-p_i)^{1-z_i}}{\theta^s \sum_{\boldsymbol{z}} \prod_{i=1}^{k} p_i^{z_i}(1-p_i)^{1-z_i}} = P\left(\boldsymbol{X} = \boldsymbol{y} \Big| \sum_{i=1}^{k} X_i = s\right)$$

där summationerna sker över alla $\boldsymbol{z}=(z_1,\ldots,z_k)$ där $z_i\in\{0,1\}$ och $\Sigma_{i=1}^{k} z_i = s$.

Den betingade sannolikhetsfunktionen för $\boldsymbol{X}$ givet $\Sigma_{i=1}^{k} X_i = s$ utan transformering kan skrivas som

$$P(\mathrm{X}=\mathrm{x}\Big|\sum_{i=1}^{k}X_i=s)=\frac{P(X_1=x_1,...,X_k=x_k)}{P(\sum_{i=1}^{k}X_i=s)}=\frac{\prod_{i=1}^{k}p_i^{x_i}(1-p_i)^{1-x_i}}{\sum_{\boldsymbol{x}}\prod_{i=1}^{k}p_i^{x_i}(1-p_i)^{1-x_i}} \qquad (1)$$

där summationen sker över alla $\boldsymbol{x}=(x_1,\ldots,x_k)$ där $x_i\in\{0,1\}$ och $\Sigma_{i=1}^{k}x_i=s$.

Man kan här se att den betingade sannolikhetsfunktionen för $\boldsymbol{Y}(\theta)$ givet $\Sigma_{i=1}^{k}Y_i(\theta)=s$ är samma som (1) för varje $\theta\in(0,\infty)$ och inte beror på θ. För att maximera sannolikheten att acceptera en vektor bör man simulera från $\boldsymbol{Y}(\theta)$ med det θ-värde som maximerar $P(\Sigma_{i=1}^{k}Y_i(\theta)=s)$. Det θ som uppfyller detta visar sig vara det som uppfyller $E\left[\Sigma_{i=1}^{k}Y_i(\theta)\right]=\Sigma_{i=1}^{k}P_i(\theta)=s$. Beteckna detta θ-värde med $\hat{\theta}$.

För att visa att $P(\Sigma_{i=1}^{k}Y_i(\hat{\theta})=s)$ maximeras för det $\hat{\theta}$ som uppfyller $\Sigma_{i=1}^{k}P_i(\hat{\theta})=s$ kan vi betrakta

$$P\left(\sum_{i=1}^{k}Y_i(\theta)=s\right)=\sum_{\boldsymbol{z}}\prod_{i=1}^{k}P_i(\theta)^{z_i}\left(1-P_i(\theta)\right)^{1-z_i}=$$

$$=\sum_{\boldsymbol{z}}\prod_{i=1}^{k}\left(\frac{p_i\theta}{1+p_i(\theta-1)}\right)^{z_i}\left(1-\frac{p_i\theta}{1+p_i(\theta-1)}\right)^{1-z_i}$$

$$=\sum_{\boldsymbol{z}}\prod_{i=1}^{k}\left(\frac{p_i\theta}{1+p_i(\theta-1)}\right)^{z_i}\left(\frac{1+p_i\theta-p_i-p_i\theta}{1+p_i(\theta-1)}\right)^{1-z_i}=$$

$$=\sum_{\boldsymbol{z}}\prod_{i=1}^{k}\left(\frac{p_i\theta}{1+p_i(\theta-1)}\right)^{z_i}\left(\frac{1-p_i}{1+p_i(\theta-1)}\right)^{1-z_i}=\sum_{\boldsymbol{z}}\prod_{i=1}^{k}\frac{p_i^{z_i}(1-p_i)^{1-z_i}\theta^{z_i}}{1-p_i+\theta p_i}=$$

$$=\frac{\theta^s}{\prod_{i=1}^{k}(1-p_i+\theta p_i)}\sum_{\boldsymbol{z}}\prod_{i=1}^{k}p_i^{z_i}(1-p_i)^{1-z_i}\ .$$

Logaritmering av $P(\Sigma_{i=1}^{k}Y_i(\theta)=s)$ ger

$$\ln P\left(\sum_{i=1}^{k}Y_i(\theta)=s\right)=\ln\left(\frac{\theta^s}{\prod_{i=1}^{k}(1-p_i+\theta p_i)}\sum_{\boldsymbol{y}}\prod_{i=1}^{k}p_i^{y_i}(1-p_i)^{1-y_i}\right)\propto$$

$$s\ln\theta-\ln\prod_{i=1}^{k}(1-p_i+\theta p_i)=s\ln\theta-\sum_{i=1}^{k}\ln(1-p_i+\theta p_i).$$

Genom att derivera med avseende på θ finner vi att

$$\frac{\partial}{\partial\theta}\ln\left(P(\Sigma_{i=1}^{k}Y_i(\theta)=s)\right)=\frac{s}{\theta}-\sum_{i=1}^{k}\frac{p_i}{1-p_i+\theta p_i}=\frac{1}{\theta}\left(s-\sum_{i=1}^{k}P_i(\theta)\right)=0,$$

dvs $P(\Sigma_{i=1}^{k}Y_i(\theta)=s)$ har en extrempunkt i det θ-värde som uppfyller $\Sigma_{i=1}^{k}P_i(\theta)=s$.

Att detta är ett maximum visas genom att studera andraderivatan

$$\frac{\partial^2}{\partial\theta^2}\ln\left(P(\Sigma_{i=1}^{k}Y_i(\theta)=s)\right)=\frac{1}{\theta^2}\left(s-\sum_{i=1}^{k}P_i(\theta)\right)-\frac{1}{\theta}\sum_{i=1}^{k}\frac{p_i^2(1-p_i)}{(1-p_i+p_i\theta)^2}<0,$$

eftersom $s-\sum_{i=1}^{k}P_i(\theta)=0$ och $\frac{1}{\theta}\sum_{i=1}^{k}\frac{p_i^2(1-p_i)}{(1-p_i+p_i\theta)^2}>0$.

Att andraderivatan är negativ visar att nollstället är ett maximum. Det betyder att $P(\Sigma_{i=1}^{k}Y_i(\theta)=s)$ maximeras för det θ som uppfyller att $\Sigma_{i=1}^{k}P(\theta)=s$.

Om man vet att $\Sigma_{i=1}^{k}p_i$ ligger en bit ifrån det *s* som är givet kan det alltså vara värt att beräkna $p_i(\hat{\theta})$, $i=1,..,k$ och byta sannolikheter. Låt oss illustrera detta i ett exempel.

Exempel 1. Antag att vi har k=5, s=3, p_1=0.1, p_2=0.2, p_3=0.3, p_4=0.4 och p_5=0.5. I det här fallet så är $\Sigma_{i=1}^{5}p_i=1.5$ vilket inte riktigt överensstämmer med $s=3$. En snabbare simulering borde i detta fall erhållas med $\boldsymbol{Y(\hat{\theta})}=(Y_1(\hat{\theta}),Y_2(\hat{\theta}),Y_3(\hat{\theta}),Y_4(\hat{\theta}),Y_5(\hat{\theta}))$ där $\hat{\theta}$ uppfyller villkoret $\Sigma_{i=1}^{k}p_i(\hat{\theta})=3$. I detta fall blir $p_1(\hat{\theta})$=0.3145, $p_2(\hat{\theta})$=0.5079, $p_3(\hat{\theta})$=0.6389, $p_4(\hat{\theta})$=0.7335, $p_5(\hat{\theta})$=0.8050 där $\hat{\theta}$=1.4181.

Om man genererar vektorn $\boldsymbol{X}$ är sannolikheten att acceptera denna 0.1274. Om man däremot genererar från $\boldsymbol{Y(\hat{\theta})}$ så är sannolikheten att acceptera vektorn knappt 0.4. Det innebär att Förkastelsemetoden med de nya sannolikheterna accepterar en vektor ungefär tre gånger så ofta, dvs den går tre gånger så snabbt.

□

3 Bondessons metod

I Bondessons metod utnyttjas definitionen av betingad sannolikhet för att simulera utfallet på X_i, $i=1,\ldots,k$. Den betingade sannolikheten för en vektor kan skrivas

$$P\left(X_1=x_1,...,X_k=x_k\left|\sum_{i=1}^{k}X_i=s\right.\right)=\prod_{j=1}^{k}P\left(X_j=x_j\left|\sum_{i=j}^{k}X_i=s-\sum_{l=0}^{j-1}x_l\right.\right),$$

där $x_0=0$.

Genom att successivt simulera utfallen från de betingade fördelningarna för $X_i \left| \Sigma_{i=j}^{k} X_i = s - \Sigma_{l=0}^{j-1} x_l \right.$ erhålls den önskade betingade fördelningen. Låt oss nu betrakta detta simuleringsförfarande mer i detalj.

Börja med att simulera ett utfall från $X_1 \left| \Sigma_{i=1}^{k} X_i = s \right.$. Med hjälp av definitionen på betingad sannolikhet kan man se att sannolikhetsfunktionen kan skrivas som

$$P\left(X_1 = x_1 \left| \sum_{i=1}^{k} X_i = s \right. \right) = \frac{P(X_1 = x_1) P\left(\sum_{i=2}^{k} X_i = s - x_1 \right)}{P\left(\sum_{i=1}^{k} X_i = s \right)}, \qquad x_1=0,1.$$

Om vi känner $P(\Sigma_{i=1}^{k} X_i = s)$ och $P(\Sigma_{i=2}^{k} X_i = s-1)$ kan vi generera ett utfall x_1. I kapitel 3.1 beskrivs hur dessa sannolikheter beräknas exakt och i kapitel 3.2 hur de kan approximeras.

Givet att utfallet på X_1 blev x_1 kan vi sedan simulera ett utfall på X_2. Enligt samma resonemang som ovan kan den betingade sannolikheten för utfallet på X_2 givet utfallet på X_1 och $\Sigma_{i=1}^{k} X_i = s$ skrivas som

$$P\left(X_2 = x_2 \left| \sum_{i=2}^{k} X_i = s - x_1 \right. \right) = \frac{P(X_2 = x_2) P\left(\sum_{i=3}^{k} X_i = s - x_1 - x_2 \right)}{P\left(\sum_{i=2}^{k} X_i = s - x_1 \right)}, \qquad x_2=0,1,$$

dvs givet att man känner utfallet på X_1 kan man i praktiken generera ett utfall på X_2. Givet utfallen på X_1 och X_2 simuleras ett utfall på X_3 osv till dess att man fått ett utfall på alla stokastiska variabler i vektorn. Det allmänna uttrycket för sannolikheterna på de olika utfallen blir

$$P\left(X_a = x_a \left| \sum_{i=a}^{k} X_i = s - \sum_{j=0}^{a-1} x_j \right. \right) = \frac{P(X_a = x_a) P\left(\sum_{i=a+1}^{k} X_i = s - \sum_{j=0}^{a} x_j \right)}{P\left(\sum_{i=a}^{k} X_i = s - \sum_{j=0}^{a-1} x_j \right)}, \quad a=1,2,\ldots,k,$$

där x_0=0.

Nu känner vi alltså den betingade sannolikhetsfunktionen för $\boldsymbol{X}$. Innan man kan använda metoden måste sannolikheterna $P(\Sigma_{i=a}^{k} X_i = t)$ bestämmas för alla *a* och *t*. Detta kan vi göra antingen exakt eller asymptotiskt. I kapitel 3.1 beskrivs hur man beräknar dessa sannolikheter exakt och i kapitel 3.2 hur de approximeras.

3.1 Exakt beräkning av sannolikheter

Det enklaste sättet att se hur $P\left(\Sigma_{i=a}^{k} X_i = t\right)$ kan beräknas är genom att studera ett exempel.

Exempel 2. Antag att vi har en vektor *(X_1, X_2, X_3, X_4, X_5)* med sannolikheter p_1=0.1, p_2=0.2, p_3=0.3, p_4=0.4, p_5=0.5 och $s = \Sigma_{i=1}^{5} X_i = 3$.

Först beräknas sannolikheterna för att $\Sigma_{i=4}^{5} X_i$ blir 0,1 eller 2. Sannolikheten för att denna summa blir 0 är *$(1-p_4)(1-p_5)$.* Sannolikheten att summan blir 2 är $p_4 p_5$. Följaktligen blir sannolikheten för att summan är 1 $p_4(1-p_5)+(1-p_4)p_5$.

Sannolikheterna för att summan $\Sigma_{i=3}^{5} X_i$ är 0 eller 3 beräknas efter samma princip som för summan med 2 termer, dvs *$(1-p_3)(1-p_4)(1-p_5)$* och $p_3 p_4 p_5$. När det gäller sannolikheterna för när $\Sigma_{i=3}^{5} X_i$ är 1 respektive 2 kan de beräknas genom att man utnyttjar tidigare beräknade sannolikheter. Sannolikheten för att summan är 1 kan skrivas som $p_3 P(\Sigma_{i=4}^{5} X_i = 0)+(1-p_3)P(\Sigma_{i=4}^{5} X_i = 1)$. På motsvarande sätt beräknas $P(\Sigma_{i=3}^{5} X_i = 2)$.

Med samma teknik kan man beräkna de övriga sannolikheterna som behövs. Tabell 1 visar de sannolikheter som behövs för det här exemplet.

□

Tabell 1: Exakta sannolikheter till Bondessons metod i Exempel 2

Händelse	Beräkning	Sannolikhet
$P(\Sigma_{i=5}^{5} X_i = 0)$	$1-p_5$	0.5
$P(\Sigma_{i=5}^{5} X_i = 1)$	p_5	0.5
$P(\Sigma_{i=4}^{5} X_i = 0)$	$(1-p_4)(1-p_5)$	0.3
$P(\Sigma_{i=4}^{5} X_i = 1)$	$p_4(1-p_5)+(1-p_4)p_5$	0.5
$P(\Sigma_{i=4}^{5} X_i = 2)$	$p_4 p_5$	0.2
$P(\Sigma_{i=3}^{5} X_i = 1)$	$p_3\, P(\Sigma_{i=4}^{5} X_i = 0)+(1-p_3)\, P(\Sigma_{i=4}^{5} X_i = 1)$	0.44
$P(\Sigma_{i=3}^{5} X_i = 2)$	$p_3\, P(\Sigma_{i=4}^{5} X_i = 1)+(1-p_3)\, P(\Sigma_{i=4}^{5} X_i = 2)$	0.29
$P(\Sigma_{i=3}^{5} X_i = 3)$	$p_3 P(\Sigma_{i=4}^{5} X_i = 2)$	0.06
$P(\Sigma_{i=2}^{5} X_i = 2)$	$p_2\, P(\Sigma_{i=3}^{5} X_i = 1)+(1-p_2)\, P(\Sigma_{i=3}^{5} X_i = 2)$	0.32
$P(\Sigma_{i=2}^{5} X_i = 3)$	$p_2\, P(\Sigma_{i=3}^{5} X_i = 2)+(1-p_2)\, P(\Sigma_{i=3}^{5} X_i = 3)$	0.106
$P(\Sigma_{i=1}^{5} X_i = 3)$	$p_1\, P(\Sigma_{i=2}^{5} X_i = 2)+(1-p_1)\, P(\Sigma_{i=2}^{5} X_i = 3)$	0.1274

Antag att ***X*** är en vektor som är *k* element lång. Om *k* är stort måste ett stort antal sannolikheter beräknas. Det maximala antalet sannolikheter av typen

$$P\left(\sum_{i=a}^{k} X_i = s - x_0 - x_1 - \ldots - x_{a-1}\right),$$

där *$a=1,\ldots,k$*, *$x_j=0,1$*, *$j=1,\ldots,k$* och *$x_0=0$*, som måste beräknas är $k^2/4+k$. Ett bevis på att detta påstående är sant finns i bilaga 2.

3.2 Approximering av sannolikheter

Exemplet i kapitel 3.1 bygger på en relativt liten vektor och det blir därför få sannolikheter som skall beräknas och det medför inga problem för en dator. När man har stora vektorer, dvs när *k* är stort, måste många sannolikhetsberäkningar utföras vilket kan innebära en stor tidsåtgång. Då kan man nöja sig med att beräkna ett mindre antal sannolikheter exakt och beräkna resterande approximativt.

Det man kan utnyttja när man ska approximera sannolikheter för summorna är att dessa summor är approximativt normalfördelade när q är stort i summan

$$S_q = \sum_{i=k-q}^{k} X_i = v_{k-q},\ 0 \le q \le \text{k}.$$

Sannolikheterna kan då approximeras som

$$P(S_q = v) \approx \frac{1}{\sqrt{2\pi\sigma_q}} \exp\left\{-\frac{(v-\mu_q)^2}{2\sigma_q^2}\right\},\ \ 0 \le q \le k,$$

där $\mu_q = \Sigma_{i=k-q}^{k} p_i$ och $\sigma_q^2 = \Sigma_{i=k-q}^{k} p_i(1-p_i)$, se t.ex. Petrov (1975)[2].

Denna approximation fungerar relativt bra om p_i är nära 0.5. För att få så bra resultat som möjligt bör man därför ordna elementen i vektorerna så att $|p_1\text{-}0.5| \le |p_2\text{-}0.5| \le .. \le |p_k\text{-}0.5|$. Det betyder att man räknar exakt på de element där sannolikheten att få 1 ligger så långt från 0.5 som möjligt.

Nu när vi kan bestämma sannolikheter både exakt och approximativt kan exempelvis $P(S_0),\ldots,P(S_{q-1})$ bestämmas exakt och $P(S_q),\ldots,P(S_k)$ bestämmas approximativt.

4 McMC

I detta kapitel presenteras en tredje metod för att generera betingade vektorer $\boldsymbol{X}$ givet $\Sigma_{i=1}^{k} X_i = s$. Denna metod benämns McMC (Markov chain Monte Carlo) och finns beskriven i t.ex. Ross (1997)[3]. Idén med McMC är att hitta en markovkedja som är lätt att simulera och vars asymptotiska fördelning är densamma som fördelningen för $\boldsymbol{X}$. Med hjälp av en sådan markovkedja kan man generera en följd av vektorer som ungefär tillhör den önskade betingade fördelningen.

4.1 Markovkedjan

En markovkedja $\{X_n, n=0,1,2,\ldots\}$ är en diskret stokastisk process med diskret tid som uppfyller markovvillkoret:

$$\text{P}[X_n=x_n|X_0=x_0, X_1=x_1,\ldots, X_{n-1}=x_{n-1}]=\text{P}[X_n=x_n|X_{n-1}=x_{n-1}]\ ,\ \ n\ge 2.$$

Att markovvillkoret är uppfyllt betyder att om man vill säga något om processens värde för X_n vid tiden n räcker det att känna till processens värde för X_{n-1} vid tiden $n-1$, dvs man får ingen extra information av att man känner värdena för X_{n-2}, X_{n-3},….

De olika värden X_n kan anta kallas tillstånd. Antag att processen befinner sig i tillståndet i vid tiden n. Oberoende av tidigare tillstånd är då sannolikheten att tillståndet är j vid tiden $n+1$ $P_{ij}= P[X_{n+1}=j|X_n=i]$. Denna sannolikhet kallas övergångssannolikhet. En matris som består av alla möjliga övergångssannolikheter kallas övergångsmatris.

En markovkedja kallas irreducibel om det för alla par av tillstånd i,j är en positiv sannolikhet att någon gång hamna i tillstånd j när man startar i tillstånd i. När $P_{ii}>0$ för något i i en irreducibel markovkedja, sägs detta tillstånd vara aperiodiskt. En markovkedja är aperiodisk om den har något aperiodiskt tillstånd. I en irreducibel markovkedja är antingen alla tillstånd aperiodiska eller alla tillstånd periodiska.

Om en irreducibel markovkedja har en stationär fördelning $\boldsymbol{\pi}=(\pi_1,\pi_2,\ldots)$, dvs om X_n har fördelningen $\boldsymbol{\pi}=(\pi_1,\pi_2,\ldots)$ vid någon tidpunkt n kommer fördelningen vara oförändrad för $n+1,n+2,\ldots$, är det en unik stationär fördelning. Om den irreducibla markovkedjan dessutom är aperiodisk så är $\boldsymbol{\pi}$ också en asymptotisk fördelning.

Existerar $\boldsymbol{\pi}=(\pi_1,\pi_2,\ldots)$ sådan att $\pi_i P_{ij}=\pi_j P_{ji}$, $\forall\ i\neq j$, sägs markovkedjan vara tidsreversibel. Man kan visa för en tidsreversibel markovkedja att $\boldsymbol{\pi}$ också är en stationär fördelning.

4.2 Idén med McMC

Antag att vi vill generera utfall från en stokastisk variabel X. Antag vidare att det är möjligt att generera tillstånd från en irreducibel aperiodisk markovkedja vars asymptotiska fördelning $\boldsymbol{\pi}$ är densamma som fördelningen för $\boldsymbol{X}$. En sådan markovkedja kan användas för att approximativt generera slumptal från den stokastiska variabeln X. Metoder för att hitta en markovkedja med dessa egenskaper finns beskrivna i kapitel 4.3 och 4.4. Om vi genererar tillstånd $x_1, x_2,\ldots$ från en sådan markovkedja, kommer $x_n,x_{n+1}\ldots$ att kunna ses som approximativa utfall från den sökta stokastiska variabeln X om n är stort. Det brukar därför rekommenderas att de första tillstånden tas bort. Hur många tillstånd som ska tas bort är svårt att veta.

4.3 Hastings-Metropolis algoritm

Låt b_j, $j=1,\ldots,m$ vara positiva tal och låt $B=\sum_{j=1}^{m} b_j$. Antag att man vill simulera en vektor med sannolikhetsfunktion $\pi_j=b_j/B$, där $j=1,\ldots,m$. Orsaken till att man ibland måste simulera π_j kan vara att m är stort eller att B är svår att beräkna.

Ett sätt att simulera π_j är att hitta en markovkedja som är relativt lätt att simulera och vars jämnviktssannolikheter är π_j. Med Hastings-Metropolis algoritm kan man konstruera en tidsreversibel markovkedja med de önskade jämnviktssannolikheterna.

Låt $\mathbf{Q}$ vara en övergångsmatris till en godtycklig irreducibel markovkedja X_n med övergångssannolikheter $q(i,j)$. $\mathbf{Q}$ bör väljas så att $q(i,j)>0$. Om $X_n=i$ genereras en stokastisk

variabel X sådan att $P(X=j)=q(i,j)$, $j=1,\ldots,m$. Antag att $X=j$. Då sätts nästa tillstånd X_{n+1} till j med någon sannolikhet $\alpha(i,j)$ och till i med sannolikhet $1-\alpha(i,j)$.

Om man fortsätter på detta sätt får man en markovkedja med övergångssannolikheter

$$P_{ij}=q(i,j)\alpha(i,j),\ j\neq i$$

$$P_{ii}=q(i,i)+\sum_{k\neq i} q(i,k)(1-\alpha(i,k)).$$

Om

$$\pi_i P_{ij}=\pi_j P_{ji},\ j\neq i$$

är denna markovkedja tidsreversibel och $\boldsymbol{\pi}=(\pi_1,\pi_2,\ldots,\pi_m)$ kommer därför också att vara den unika stationära fördelningen. Eftersom $P_{ij}=q(i,j)\alpha(i,j)$, $\forall\ j\neq i$, gäller detta om

$$\pi_i q(i,j)\alpha(i,j)=\pi_j q(j,i)\alpha(j,i),\ j\neq i,$$

dvs om

$$\alpha(i,j)=\min\left(\frac{\pi_j q(j,i)}{\pi_i q(i,j)},1\right)=\min\left(\frac{b_j q(j,i)}{b_i q(i,j)},1\right).$$

Fördelningen $\boldsymbol{\pi}$ kommer dessutom att vara den asymptotiska fördelningen till markovkedjan om den är aperiodisk, dvs om $P_{ii}>0$ för något i.

Hastings-Metropolis algoritm kan sammanfattas på följande sätt.

1. Välj en godtycklig irreducibel markovkedja med övergångsmatris **Q**. Välj också ett heltal i mellan 1 och m.
2. Låt $n=0$ och $X_0=i$.
3. Generera en stokastisk variabel X sådan att $P(X=j)=q(X_n,j)$ och ett slumptal U.
4. Om $U<[b_X q(X,X_n)]/[b_{Xn} q(X_n,X)]$, sätt $Y=X$ annars sätt $Y=X_n$.
5. $n:=n+1$ och $X_n=Y$.
6. Gå till 3.

4.4 Gibbs sampler

I simuleringsstudien i denna rapport används ett specialfall av Hastings-Metropolis algoritm som benämns Gibbs sampler. Det är också den mest använda versionen av Hastings-Metropolis algoritm.

Låt $\boldsymbol{X}= (X_1,\ldots,X_n)$ vara en stokastisk vektor med sannolikhetsfunktion p($\boldsymbol{x}$). Antag att man vill generera stokastiska vektorer med den betingade fördelningen av $\boldsymbol{X}$ givet att $\boldsymbol{X}\in A$, där A är någon mängd. En vektor med denna betingade fördelning får då sannolikhetsfunktionen

$$\mathrm{f}(\mathbf{x})=\frac{p(\mathbf{x})}{P(\mathbf{X}\in A)} \quad , \ \boldsymbol{x}\in A.$$

Gibbs sampler fungerar på följande sätt. Välj ut en av variablerna $X_1,\ldots,X_n$. Alla variabler ska ha lika stor chans att bli vald. Antag att variabel X_i väljs och att vi kan generera ett utfall X från den betingade stokastiska variabeln X_i givet $X_1=x_1,\ldots, X_{i-1}=x_{i-1}, X_{i+1}=x_{i+1},\ldots, X_n=x_n$. Om $X=x$ så låt $\boldsymbol{y}= (x_1,\ldots,x_{i-1},x,x_{i+1},\ldots,x_n)$.

Man kan se att Gibbs sampler är ett specialfall av Hastings-Metropolis algoritm med

$$\mathrm{q}(\mathbf{x},\mathbf{y})=\frac{1}{n}\frac{p(\mathbf{y})}{P(X_j=x_j,\, j\neq i)}.$$

Den nya vektorn $\boldsymbol{y}$ accepteras med sannolikheten

$$\alpha(\mathbf{x},\mathbf{y})=\min\left(\frac{f(\mathbf{y})q(\mathbf{y},\mathbf{x})}{f(\mathbf{x})q(\mathbf{x},\mathbf{y})},1\right),$$

där

$$\frac{f(\mathbf{y})q(\mathbf{y},\mathbf{x})}{f(\mathbf{x})q(\mathbf{x},\mathbf{y})}=\frac{f(\mathbf{y})p(\mathbf{x})}{f(\mathbf{x})p(\mathbf{y})}=1,$$

dvs vi kommer alltid att acceptera den nya vektorn. På samma sätt som Hastings-Metropolis algoritm kan Gibbs sampler sammanfattas i ett antal steg.

1. Låt $\boldsymbol{x}=(x_1,\ldots,x_n)$ vara en vektor där p($\boldsymbol{x}$)>0.
2. Generera ett slumptal i från en diskret likformig stokastisk variabel definierad på $\{1,\ldots,n\}$.
3. Generera ett tal x från den betingade fördelningen för X_i givet $X_j=x_j$, $j\neq i$.
4. Sätt $X_i=x$
5. Gå till 2.

Exempel 3. Antag att vi vill simulera samma vektorer som i exempel 2. Dvs antag att vi har en vektor $(X_1, X_2, X_3, X_4, X_5)$ med sannolikheter p_1=0.1, p_2=0.2, p_3=0.3, p_4=0.4, p_5=0.5 och s=3.

Om man vill använda sig av McMC i denna situation är det lämpligt att använda sig av Gibbs sampler. Man kan emellertid inte använda Gibbs sampler som den är beskriven tidigare. Om man ändrar ett element i vektorn förändras s och villkoret s=3 är ej uppfyllt. För att lösa detta

problem tas fler än ett element ut ur i vektorn. Övergångssannolikheter beräknas sedan för övergångar till de kombinationer av element som uppfyller betinget.

Det första man gör är att utse en startvektor. För stora vektorer kan det vara värt att utse den på något sätt där man tar hänsyn till fördelningen. I detta fall är dock vektorn så pass liten att startvektorn kan tas ut på ett mer godtyckligt sätt. I detta fall har startvektorn valts som ***x****=(1,0,1,0,1)*.

Nu skall ett antal element väljas ut. I detta fall tas två element ut i varje grupp av element, och vi benämner detta förfarande parvis Gibbs sampler. Det är även möjligt att ta med fler än två element i varje grupp. Vilka element som skall tas ut kan bestämmas på olika sätt. Det viktiga är att i längden ska varje element ha lika stor sannolikhet att bli vald. I detta exempel har elementen valts ut slumpvis.

Antag att X_1 och X_2 valdes i den första iterationen. Eftersom X_1=1 och X_2=0 så finns två kombinationer att välja på för att fortfarande behålla villkoret att $\Sigma_{i=1}^{5} X_i = 3$. Dessa möjligheter är att behålla vektorn som den är, dvs X_1=1 och X_2=0, eller sätta X_1=0 och X_2=1.

Sannolikheten att behålla samma vektor är

$$P(X_1 = 1, X_2 = 0) = \frac{p_1(1-p_2)}{p_1(1-p_2)+(1-p_1)p_2} = \frac{0.1*0.8}{0.1*0.8+0.9*0.2} \approx 0.3,$$

vilket betyder att sannolikheten att byta vektor i detta fall blir

$$P(X_1=0, X_2=1) \approx 0.7.$$

Antag att simuleringen ger att vi väljer X_1=0 och X_2=1. Det innebär att man får följande nya vektor ***x****=(0,1,1,0,1)*.

Antag att nästa par av stokastiska variabler som valdes ut blev X_2 och X_3. Här ser man att båda elementen har värdet 1. Det innebär att vektorn förblir densamma med sannolikhet 1 ty annars skulle villkoret $\Sigma_{i=1}^{5} X_i = 3$ ej vara uppfyllt. På detta sätt fortsätter man nu att simulera till dess att man fått önskat antal vektorer som är av den betingade fördelningen. Som det nämndes i kapitel 4.2 bör ett antal vektorer i början förkastas till dess att den approximativa fördelningen har stabiliserats.

□

5 Simulering

I denna simulering jämförs McMC-metoden, Bondessons metod och Förkastelsemetoden. Det som simuleras är betingade sannolikheter

$$P\left[\sum_{i=1}^{k} iX_i > t \middle| \sum_{i=1}^{k} X_i = s\right], \text{ för givna värden på } t \text{ och } s. \qquad (2)$$

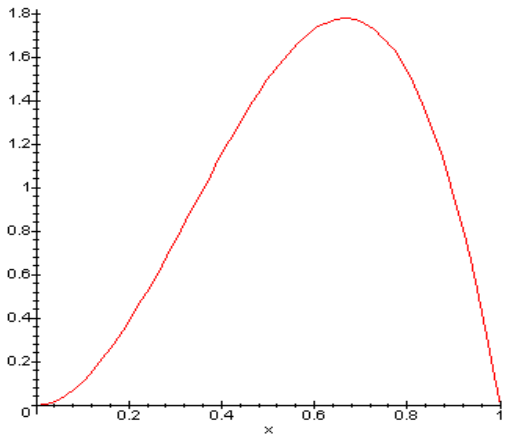
Alla program till simuleringarna är skrivna i Pascal på PC. Eftersom Pascal inte kan lagra stora datamängder i matriser, valdes att simulera vektorer $\boldsymbol{X}=(X_1,\ldots,X_k)$ med k=80. För alla tre metoderna har s valts till värdena 20, 40 respektive 60.

Sannolikheterna $p_1,\ldots,p_k$, där $p_i=P(X_i=1)$, simuleras från en *Beta(3,2)*-fördelning vars frekvensfunktion är avbildad i Figur 1 och ges av

$$f(x) = \frac{\Gamma(5)}{\Gamma(3)\Gamma(2)} x^2(1-x), \ 0 < x < 1,$$

där $\Gamma(p)$ är gammafunktionen som ges av

$$\Gamma(p) = \int_0^\infty x^{p-1}e^{-x}dx, \ p > 0.$$



Figur 1: Frekvensfunktionen för en Beta(3,2)-fördelning

För alla tre metoderna skattas sannolikheten (2) med ett medelvärde av 1000 observationer. Dessa skattningar upprepas 1000 gånger, dvs vi får 1000 skattningar av (2) från varje metod. Standardavvikelser beräknas på dessa 1000 skattningar. Vidare har konstanten t i (2) valts så att andelen medelvärden som är större än detta värde är ungefär 5%, eftersom sådana sannolikheter kan vara svåra att beräkna exakt. Fördelningsfunktioner som kan se ganska likartade ut kan ha märkbara skillnader i ytterområdet på fördelningen. För varje s testades tre olika t-värden betecknade t_1, t_2 samt t_3. Efter att ha provat några olika värden på t så användes värdena i tabell 2 som kan betraktas som lämpliga.

Tabell 2: Val av värden på t_1, t_2 och t_3 i simuleringen

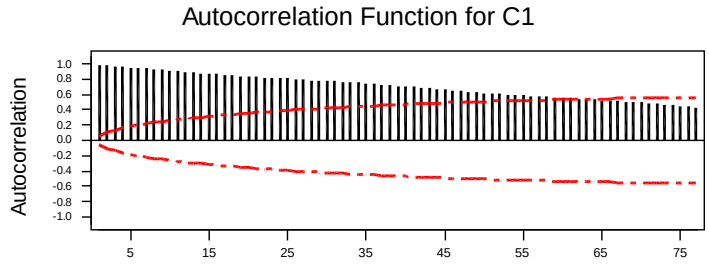
	t_1	t_2	t_3
s=20	850	875	900
s=40	1675	1700	1725
s=60	2500	2525	2550

5.1 Simulering med McMC

I undersökningen har resultat av de möjliga kombinationerna mellan *s* och *t* simulerats. Vidare undersöks skillnaden mellan att ha två eller tre element i varje grupp där elementen kan ändras. Det dessa olika metoder benämns parvis Gibbs sampler samt trippelvis Gibbs sampler. I både parvis och trippelvis Gibbs sampler väljs elementen som ska förändras vid varje iteration ut slumpvis.

I McMC finns ett starkt beroende mellan två i tiden närliggande tillstånd inom ett medelvärde vilket följande exempel visar.

Exempel 4: Skillnader i autokorrelationsfunktion med olika avstånd mellan obs.



Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ
1	0.99	31.32	983.97	16	0.87	5.27	1.4E+04	31	0.77	3.55	2.4E+04	46	0.66	2.61	3.2E+04	61	0.56	2.04	3.8E+04
2	0.98	18.04	1952.10	17	0.86	5.09	1.5E+04	32	0.77	3.48	2.5E+04	47	0.65	2.55	3.3E+04	62	0.55	2.01	3.8E+04
3	0.97	13.94	2906.92	18	0.85	4.92	1.5E+04	33	0.76	3.41	2.5E+04	48	0.64	2.50	3.3E+04	63	0.54	1.98	3.9E+04
4	0.97	11.73	3846.68	19	0.85	4.77	1.6E+04	34	0.76	3.34	2.6E+04	49	0.63	2.45	3.4E+04	64	0.54	1.95	3.9E+04
5	0.96	10.30	4771.06	20	0.84	4.63	1.7E+04	35	0.75	3.28	2.7E+04	50	0.62	2.41	3.4E+04	65	0.53	1.92	3.9E+04
6	0.95	9.27	5680.96	21	0.84	4.50	1.8E+04	36	0.74	3.21	2.7E+04	51	0.61	2.37	3.4E+04	66	0.52	1.89	4.0E+04
7	0.94	8.49	6576.09	22	0.83	4.38	1.8E+04	37	0.73	3.14	2.8E+04	52	0.61	2.32	3.5E+04	67	0.52	1.85	4.0E+04
8	0.93	7.87	7456.38	23	0.82	4.26	1.9E+04	38	0.73	3.08	2.8E+04	53	0.60	2.29	3.5E+04	68	0.51	1.82	4.0E+04
9	0.93	7.36	8321.58	24	0.82	4.16	2.0E+04	39	0.72	3.01	2.9E+04	54	0.59	2.25	3.6E+04	69	0.50	1.79	4.0E+04
10	0.92	6.92	9171.77	25	0.81	4.06	2.0E+04	40	0.71	2.95	2.9E+04	55	0.59	2.22	3.6E+04	70	0.49	1.76	4.1E+04
11	0.91	6.55	1.0E+04	26	0.81	3.96	2.1E+04	41	0.70	2.89	3.0E+04	56	0.58	2.19	3.6E+04	71	0.49	1.72	4.1E+04
12	0.90	6.23	1.1E+04	27	0.80	3.87	2.2E+04	42	0.69	2.83	3.0E+04	57	0.58	2.16	3.7E+04	72	0.48	1.69	4.1E+04
13	0.89	5.94	1.2E+04	28	0.79	3.78	2.2E+04	43	0.68	2.77	3.1E+04	58	0.57	2.13	3.7E+04	73	0.47	1.65	4.1E+04
14	0.88	5.70	1.2E+04	29	0.79	3.70	2.3E+04	44	0.67	2.72	3.1E+04	59	0.57	2.10	3.7E+04	74	0.46	1.62	4.2E+04
15	0.88	5.47	1.3E+04	30	0.78	3.63	2.4E+04	45	0.66	2.66	3.2E+04	60	0.56	2.07	3.8E+04	75	0.45	1.58	4.2E+04

Figur 2 Autokorrelationsfunktion med varje vektor medräknad

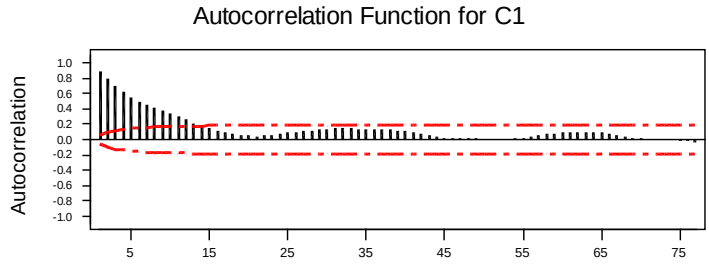
Figur 2 visar autokorrelationsfunktionen av värdena på för 1000 på varandra följande tillstånd, dvs varje simulerad vektor är medräknad. I detta exempel gäller att vektorn är 80 termer lång. Simuleringsmetoden som används är parvis Gibbs sampler med *s*=40. De 10000 första omkastningarna har förkastats för att man skall ”vara nära” den rätta fördelningen. Man kan se en stor korrelation mellan i tiden närliggande tillstånd.

Detta stora positiva beroende mellan de i tiden närliggande tillstånden gör att resultaten blir osäkra, vi får en stor varians. Man vill helst att tillstånden ska vara helt oberoende men detta är dock svårt att uppnå. För att bättre uppnå oberoende kan man exempelvis ta med var tionde eller var hundrade simulerad vektor i skattningen. Om man tar med var hundrade vektor kommer simuleringen att ta betydligt längre tid men man får samtidigt en skattning med mindre varians vilket ger en bättre skattning.

Nästa fråga blir då att avgöra hur långt mellanrum man skall ha mellan varje medtagen observation. I den här simuleringen så har autokorrelationsfunktionen studerats.

Fall 1:

I fall 1 används parvis Gibbs sampler.



Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ
1	0.89	28.28	802.15	16	0.12	1.28	3829.77	31	0.15	1.55	3966.14	46	0.01	0.14	4152.05	61	0.09	0.95	4190.33
2	0.80	15.59	1436.89	17	0.09	1.01	3838.69	32	0.14	1.53	3987.72	47	0.01	0.14	4152.25	62	0.09	0.92	4198.68
3	0.71	11.36	1937.92	18	0.08	0.81	3844.44	33	0.14	1.52	4009.13	48	0.02	0.19	4152.60	63	0.09	0.91	4207.01
4	0.63	9.02	2336.02	19	0.06	0.65	3848.13	34	0.14	1.48	4029.50	49	0.02	0.20	4153.01	64	0.09	0.95	4216.10
5	0.56	7.40	2647.34	20	0.05	0.53	3850.64	35	0.14	1.47	4049.77	50	0.01	0.08	4153.07	65	0.09	0.98	4225.71
6	0.50	6.31	2898.74	21	0.05	0.50	3852.90	36	0.14	1.47	4070.03	51	0.00	0.04	4153.09	66	0.08	0.84	4232.79
7	0.46	5.56	3109.77	22	0.05	0.53	3855.41	37	0.14	1.44	4089.70	52	-0.00	-0.02	4153.09	67	0.06	0.63	4236.74
8	0.42	4.97	3289.21	23	0.06	0.64	3859.07	38	0.13	1.37	4107.40	53	0.00	0.02	4153.09	68	0.04	0.42	4238.49
9	0.38	4.42	3438.32	24	0.07	0.79	3864.57	39	0.12	1.24	4122.07	54	0.01	0.15	4153.31	69	0.02	0.23	4239.01
10	0.34	3.85	3555.63	25	0.09	0.96	3872.86	40	0.11	1.11	4133.82	55	0.02	0.25	4153.89	70	0.01	0.12	4239.15
11	0.30	3.30	3644.48	26	0.10	1.08	3883.37	41	0.09	0.90	4141.59	56	0.04	0.45	4155.85	71	0.00	0.01	4239.15
12	0.26	2.84	3712.14	27	0.11	1.15	3895.34	42	0.07	0.73	4146.76	57	0.06	0.61	4159.47	72	-0.01	-0.06	4239.19
13	0.22	2.38	3760.17	28	0.12	1.26	3909.53	43	0.05	0.56	4149.81	58	0.07	0.76	4165.19	73	-0.01	-0.08	4239.26
14	0.18	1.97	3793.62	29	0.12	1.32	3925.40	44	0.04	0.42	4151.48	59	0.08	0.84	4172.17	74	-0.01	-0.06	4239.30
15	0.15	1.59	3815.55	30	0.14	1.44	3944.33	45	0.02	0.20	4151.88	60	0.09	0.96	4181.32	75	-0.02	-0.16	4239.55

Figur 3 Autokorrelationsfunktion för parvis Gibbs sampler med var tionde vektor medräkand.

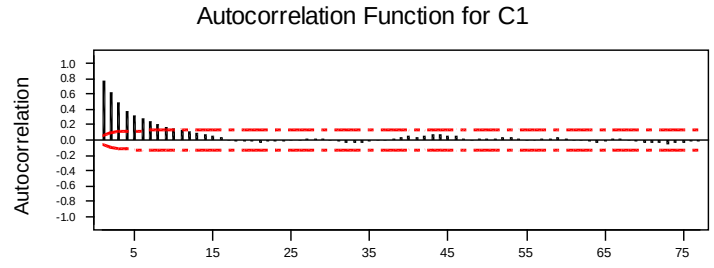
I figur 3 gäller samma förutsättningar som i figur 2 förutom att i figur 3 så visar autokorrelationsfunktionen när var tionde observation är medräknad. Även i figur 3 så är de 10000 första observationerna borttagna.

I figur 3 kan man avläsa att om man vill ligga inom ett 95% konfidensintervall, dvs T<1.96, så krävs att man väljer lag 15 eller högre. Eftersom varje ”lag” representerar 10 omkastningar krävs att man har 150 omkastningar mellan varje medtagen observation.

Denna metod för att bestämma hur många omkastningar som skall göras mellan varje medräknad observation är ganska osäker eftersom resultatet kan variera lite mellan olika körningar och mellan situationer som studeras. Man får ändå en ganska bra uppskattning av hur långt mellanrum mellan observationerna man skall ha så det kan vara värt att genomföra detta test ändå trots en viss osäkerhet.

Fall 2:

I simuleringarna finns också trippelvis Gibbs sampler där i övrigt samma förutsättningar gäller som i fall 1. Figur 4 visar autokorrleationsfunktionen för fall 2.



Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ	Lag	Corr	T	LBQ
1	0.78	24.80	617.05	16	0.03	0.44	1776.02	31	-0.02	-0.29	1780.99	46	0.05	0.73	1813.98	61	0.00	0.07	1820.07
2	0.62	13.11	1002.03	17	0.00	0.05	1776.03	32	-0.03	-0.51	1782.23	47	0.02	0.36	1814.60	62	-0.01	-0.11	1820.13
3	0.49	8.98	1245.21	18	-0.02	-0.27	1776.35	33	-0.05	-0.70	1784.51	48	0.01	0.11	1814.66	63	-0.03	-0.39	1820.89
4	0.39	6.58	1396.90	19	-0.02	-0.31	1776.81	34	-0.03	-0.45	1785.45	49	0.01	0.16	1814.78	64	-0.03	-0.47	1821.99
5	0.32	5.24	1501.67	20	-0.02	-0.31	1777.26	35	-0.02	-0.35	1786.01	50	0.02	0.23	1815.04	65	-0.01	-0.17	1822.13
6	0.28	4.41	1580.01	21	-0.03	-0.43	1778.10	36	-0.00	-0.03	1786.02	51	0.02	0.33	1815.56	66	0.01	0.17	1822.28
7	0.24	3.69	1636.93	22	-0.03	-0.37	1778.75	37	0.01	0.08	1786.05	52	0.03	0.46	1816.57	67	0.01	0.15	1822.39
8	0.20	3.05	1677.07	23	-0.01	-0.16	1778.87	38	0.02	0.23	1786.30	53	0.03	0.47	1817.67	68	-0.00	-0.04	1822.39
9	0.17	2.62	1707.26	24	-0.01	-0.20	1779.06	39	0.03	0.51	1787.55	54	0.02	0.26	1818.00	69	-0.02	-0.33	1822.93
10	0.15	2.25	1729.85	25	-0.00	-0.02	1779.06	40	0.05	0.78	1790.46	55	0.00	0.06	1818.02	70	-0.04	-0.57	1824.58
11	0.13	1.94	1746.73	26	0.01	0.10	1779.11	41	0.05	0.68	1792.66	56	0.00	0.06	1818.04	71	-0.04	-0.52	1825.91
12	0.11	1.64	1758.96	27	0.02	0.23	1779.36	42	0.06	0.88	1796.35	57	0.02	0.23	1818.30	72	-0.05	-0.69	1828.32
13	0.09	1.34	1767.13	28	0.03	0.39	1780.06	43	0.07	1.03	1801.43	58	0.02	0.29	1818.71	73	-0.05	-0.80	1831.53
14	0.07	1.01	1771.80	29	0.02	0.32	1780.54	44	0.08	1.14	1807.65	59	0.03	0.46	1819.76	74	-0.04	-0.57	1833.19
15	0.06	0.85	1775.11	30	0.01	0.10	1780.59	45	0.06	0.88	1811.41	60	0.02	0.24	1820.05	75	-0.03	-0.43	1834.10

Figur 4 Autokorrelationsfunktion för trippelvis Gibbs sampler med var tionde vektor medräknad.

Om man även i fall 2 vill ha 95%-igt konfidensintervall, ser man att man bör ta ”lag” 10 vilket betyder att man bör ha ungefär 100 omkastningar mellan varje medtagen observation i detta fall. Man kan se att det krävs färre omkastningar i detta fall än för parvis Gibbs sampler. Detta kan bero på att man i trippelvis Gibbs sampler får en större variation i varje omkastning vilket bör betyda att man tidigare uppnår ett oberoende mellan vektorerna.

5.2 Simulering med Bondessons metod

För att kunna använda Bondessons metod måste man först beräkna sannolikheter av typen

$$P\left(\sum_{i=a}^{k} X_i = s - x_0 - x_1 - \ldots - x_{a-1}\right),\ a=1,\ldots,k,\ x_j=0,1,\ j=1,\ldots,k,\ x_0=0,$$

som också beskrevs i kapitel 3.

Dessa beräkningar är relativt tidskrävande men behöver bara göras en gång. I pascalprogrammen för Bondessons metod har fler sannolikheter än nödvändigt beräknats beroende på att programmen blev mer effektiva på detta sätt. Om man bara vill beräkna de nödvändiga summorna kan man t.ex. lösa detta med ett relativt stort antal if-satser vilket gör att programmet kan bli ganska stort och ”trögt”.

Om man approximerar en del sannolikheter bör man innan dessa beräkningar utförs ha ordnat vektorn så att $|p_1\text{-}0.5| \leq |p_2\text{-}0.5| \leq \ldots \leq |p_k\text{-}0.5|$. Detta utförs som tidigare nämnts för att beräkna exakt på de termer där sannolikheten att få 1 är så långt från 0.5 som möjligt.

När det gäller approximation med hjälp av normalfördelningsantagande, har simuleringen i denna rapport delats in i två grupper för att undersöka tidsåtgång och precision i simuleringen. I den ena gruppen beräknas alla sannolikheter exakt och i den andra gruppen beräknas

sannolikheterna för 40, dvs hälften, av termerna exakt medan sannolikheterna för de resterande termerna approximeras.

6 Resultat

Resultaten från simuleringen finns redovisade i bilaga 1.

Nedanstående beteckningar kommer att användas i följande text och i bilaga 1.

E : skattad andel observationer över t , dvs medelvärde av skattningar.
SD : Standardavvikelsen för de medelvärden som använts för att beräkna E.
Tid : Tiden som den aktuella simuleringen tog i sekunder (CPU-tid).
e : antal termer i den del där sannolikheterna för de olika summorna beräknats exakt i Bondessons metod.
b : antal i det block av termer där omkastning kan inträffa i McMC.

I simuleringen har *t* valts så att E skall ligga i närheten av 0.05. När normalapproximation använts i Bondessons metod kan man se att E avviker från de andra metoderna. I en jämförelse med de andra metoderna ser man att väntevärdena har skattats för lågt. Det kan bero på att normalapproximationen i detta fall ger en för grov skattning. Det finns dock metoder som ger bättre skattningar, se t.ex. avsnittet om rejektiv sampling i Hajek[4]. För de övriga metoderna kan man se att E ligger väldigt nära varandra om *s* och *k* är lika. De små skillnader som finns beror troligtvis på slumpfel.

Man kan också se skillnader i standardavvikelsen SD. Förkastelsemetoden och Bondessons metod ger ungefär samma värden på SD och där får de små skillnaderna betraktas som slumpfel. När det gäller McMC kan man dock se att SD är betydligt högre än för de två andra metoderna. Detta gäller speciellt parvis Gibbs sampler. Man kan se en viss förhöjning av värdena i SD även för trippelvis Gibbs sampler men det är inte alls lika tydligt som för parvis Gibbs sampler.

Värdena på SD för Förkastelsemetoden och för Bondessons metod är relativt lika så frågan är nu varför man får för höga värden på SD i McMC. Det beror troligtvis på att man inte fått bort allt beroende mellan vektorerna fast än man har förkastat att stort antal observationer mellan varje medtagen observation. I praktiken så kommer man i McMC alltid att ha ett litet beroende mellan observationerna. För att få bort beroendet helt måste man förkasta ”oändligt” många observationer mellan varje medtagen observation. Att förhöjningen är störst för parvis Gibbs sampler beror på att man på detta sätt inte får lika bra variation mellan de i tiden närliggande vektorerna som när fler termer ingår i varje omkastningsblock.

Beträffande tidsaspekten kan man se att simuleringarna med Förkastelsemetoden och Bondessons metod går på acceptabel tid och av dessa två verkar Bondessons metod gå snabbast, med reservation för programmeringen. McMC kräver längre tid. Detta gäller speciellt trippelvis Gibbs sampler då McMC tar betydligt längre tid än de andra metoderna.

Det finns också tidsskillnader inom varje metod. I Förkastelsemetoden och McMC kan man se markanta tidsskillnader där metoderna tar längre tid på sig när *s*=40. Man kan mellan dessa

två metoder också se små skillnader mellan s=20 och s=60. Dessa skillnader är relativt små om man jämför med skillnaden till s=40 men de är ända så tydliga att de inte verkar bero på slumpfel.

Skillnaderna i tidsåtgång följer alltså samma mönster i Förkastelsemetoden och i McMC. Däremot bryts detta mönster ganska tydligt av Bondessons metod där simuleringen verkar gå snabbare ju högre värde på s man har.

En sak som är överraskande med Bondessons metod är att det inte verkar spela någon större roll om alla sannolikheter för de olika summorna beräknas exakt eller om en del beräknas med hjälp av normalapproximation. Möjligtvis kan man sänka tidsåtgången med normalapproximation om s=60 men det är marginellt.

En orsak till att Bondessons metod i detta fall inte går snabbare med normalapproximation kan vara att det är en relativt liten simulering. Detta innebär att beräkningstiderna för sannolikheterna på de olika summorna är små oavsett om man normalapproximerar eller ej. Skillnaden i tidsåtgång mellan metoderna är därmed marginell.

7 Slutsats

Till att börja med kan man se att alla tre metoderna verkar ge likartade skattningar. Man kan dock se att McMC verkar vara en mindre bra metod, jämfört med de andra, för den här sortens simulering, ty den tar betydligt längre tid än de övriga metoderna och skattningarna får något större standardavvikelser. Det beror troligtvis på att det finns ett beroende mellan i tiden närliggande vektorer i McMCsimuleringen. Om man vill ha en simuleringstid som är någorlunda jämförbar med Förkastelsemetoden och Bondessons metod, krävs att man använder parvis Gibbs sampler. Detta betyder dock att beroendet mellan i tiden närliggande vektorer blir större. Att vi har detta beroende mellan i tiden närliggande vektorer betyder att McMC i detta fall ger ett högre värde på SD än Förkastelsemetoden och Bondessons metod.

Om man vill ha bättre resultat från McMC kan man använda trippelvis Gibbs sampler. Det betyder dock en avsevärt ökad simuleringstid vilket får ses som en stor nackdel speciellt när man trots trippelvis Gibbs sampler ändå inte får lika bra resultat som med Förkastelsemetoden och Bondessons metod. Man skulle kunna pröva med fler termer än tre i varje omkastningsgrupp men det har inte gjorts i denna simuleringsstudie. Troligtvis skulle man se ett bättre resultat än för trippelvis Gibbs sampler men få en betydligt ökad simuleringstid.

Simuleringstiden behöver inte bara bero på hur snabba metoderna är utan kan också bero på hur programmen är skrivna. Man kan ha optimerat ett program på ett bra sätt och gjort något annat program lite sämre. Man kan dock trots det se tendenser till vilken metod som är snabbast så det kan vara intressant att studera dessa tider ändå.

När det gäller att bestämma vilken av Förkastelsemetoden eller Bondessons metod som passar bäst verkar båda vara ganska bra. Utan normalapproximation ser Bondessons metod liksom Förkastelsemetoden ut att skatta likartat. Standardavvikelserna SD är i samma storlek för båda metoderna. Fördelen med Bondessons metod är att simuleringen går snabbare än med Förkastelsemetoden. Däremot är Förkastelsemetoden enklare att implementera i ett program så vilken metod av dessa man väljer är en smaksak. Personligen skulle jag ha valt Bondessons

metod eftersom jag tycker att den relativt stora tidsvinsten i simuleringen uppväger besväret med den krångligare implementeringen. Man skall dock tänka på att Bondessons metod i det ena fallet är approximativ.

Tack

Jag skulle vilja tacka min handledare Leif Nilsson för hans hjälp och engagemang under skrivandet av detta examensarbete. Jag vill också tacka Professor Lennart Bondesson för värdefulla råd och idéer under arbetets gång.

Referenser

[1] Broström, G och Nilsson, L. (1999). Acceptance-rejection sampling from the conditional distribution of independent discrete random variables, given their sum. Insänd till *Statistics*

[2] Petrov, V.V. (1975). *Sums of independent random variables*. Springer-Verlag, New York.

[3] Ross, S.M. (1997). *Simulation*, second edition. Academic Press, New York.

[4] Hajek, J. (1981). *Sampling from a finite population*. Marcel Dekker, New York.

Bilaga 1

Simuleringsresultat

Förkastelsemetoden

s	t	E	SD	Tid (sekunder)
20	850	0.13231	0.01075	3390.596
20	875	0.07901	0.00865	3387.497
20	900	0.04389	0.00653	3391.774
40	1675	0.10663	0.00963	3852.408
40	1700	0.06739	0.00788	3831.890
40	1725	0.03962	0.00621	3849.470
60	2500	0.08693	0.00917	3428.875
60	2525	0.04894	0.00672	3429.769
60	2550	0.02578	0.00481	3421.905

Bondessons metod

s	t	e	E	SD	Tid (sekunder)
20	850	80	0.13285	0.01072	2328.875
20	875	80	0.07901	0.00835	2327.958
20	900	80	0.04374	0.00626	2330.536
40	1675	80	0.10734	0.01007	2275.589
40	1700	80	0.06711	0.00775	2276.809
40	1725	80	0.03981	0.00609	2277.430
60	2500	80	0.08622	0.00921	2216.697
60	2525	80	0.04921	0.00681	2215.591
60	2550	80	0.02613	0.00513	2217.690
20	850	40	0.11325	0.00979	2330.440
20	875	40	0.06522	0.00794	2332.981
20	900	40	0.03429	0.00559	2329.405
40	1675	40	0.10563	0.01017	2276.727
40	1700	40	0.06567	0.00801	2280.643
40	1725	40	0.03889	0.00592	2278.283
60	2500	40	0.08340	0.00873	2208.805
60	2525	40	0.04742	0.00677	2209.927
60	2550	40	0.02478	0.00499	2210.449

McMC metoden

s	t	b	E	SD	Tid (sekunder)
20	850	2	0.13218	0.01169	4198.829
20	875	2	0.07900	0.00912	4198.081
20	900	2	0.04381	0.00681	4197.295
40	1675	2	0.10689	0.01047	5132.787
40	1700	2	0.06726	0.00886	5133.547
40	1725	2	0.03952	0.00627	5132.407
60	2500	2	0.08668	0.00969	4195.210
60	2525	2	0.04932	0.00735	4193.890
60	2550	2	0.02596	0.00528	4194.687
20	850	3	0.13248	0.01074	11959.557
20	875	3	0.07956	0.00852	11960.161
20	900	3	0.04374	0.00655	11951.530
40	1675	3	0.10718	0.01004	15199.845
40	1700	3	0.06762	0.00817	15202.686
40	1725	3	0.03971	0.00627	15204.667
60	2500	3	0.08639	0.00918	11366.749
60	2525	3	0.04896	0.00686	11369.520
60	2550	3	0.02601	0.00518	11382.575

Bilaga 2

Beräkning av antalet sannolikheter till Bondessons metod

Om $a=k$ kan man få två summor, 0 respektive 1. Vidare om $a=k-1$ så kan man få tre olika summor, 0, 1 och 2. På detta sätt ökar antalet summor till dess att $a=k-s+1$ då man kan få s olika värden på summorna $(0, 1,..., s-1)$. Detta innebär att totala antal summor i denna del är

$$\sum_{i=1}^{s}(i+1)=\frac{s(s+1)}{2}+s\,.$$

Om man börjar från andra hållet så ser man att när $a=1$ så kan man få en summa (s). När $a=2$ kan man få två summor (s och $s-1$). På detta sätt kan man resonera till dess att $a=s$ då man kan få s summor ($1, \ldots, s$). Detta innebär att totalt antal summor i denna andra del är

$$\sum_{i=1}^{s} i=\frac{s(s+1)}{2}.$$

Nu har man kvar de summor man kan få när $s<a<k-s+1$. I detta område kan man se att för varje a finns $s+1$ summor. Eftersom det i detta område finns $k-2s$ olika a så blir det totala antalet summor i detta område $(k-s)(s+1)$.

Det totala antalet summor som man måste beräkna sannolikheter för blir alltså

$$\frac{s(s+1)}{2}+s+\frac{s(s+1)}{2}+(k-s)(s+1)=s(s+1)+s+(k-s)(s+1)=k(s+1)-s^2\,.$$

$k(s+1)-s^2$ kan bli ganska stort om man har ett stort k. För givet k är det maximala antalet sannolikheter som måste beräknas $s=k/2$.

När $s=k/2$ gäller att

$$k(s+1)-s^2=k\left(\frac{k}{2}+1\right)-\frac{k^2}{2^2}=\frac{k^2}{2}+k-\frac{k^2}{4}=\frac{k^2}{4}+k\,.$$

Dvs det maximala antalet sannolikheter som måste beräknas är $k^2/4+k$.