WORKING PAPERS ISSN 0781-4410

SUOMEN AKTUAARIYHDISTYS
The Actuarial Society of Finland

7

Raimo Voutilainen

POISSON-GENERAATTORIT (1983)

6

POISSON-GENERAATTORIT

SHV-harjoitustyö
Raimo Voutilainen
Eläkevakuutusosakeyhtiö Ilmarinen
9.9.1983

SISÄLLYS

1. Johdanto

Poisson(μ)-jakauman pistetodennäköisyysfunktio on

$$(1) \qquad p_k(\mu) = \frac{e^{-\mu}\mu^k}{k!}, \quad k = 0, 1, 2, \ldots ,$$

jossa parametri μ on positiivinen reaaliluku.

Poisson-jakauma on yksi tunnetuimmista ja eniten käytetyistä diskreeteistä todennäköisyysjakaumista. Erinomaisessa käsikirjassaan [6] Haight luettelee joitakin sen sovellusalueita: teollisuus, maatalous ja ympäristötutkimus, biologia, lääketiede, puhelinliikenne, onnettomuudet, kauppa, jonoteoria, sosiologia ja väestötiede, liikenteenohjaus, sotatiede, hiukkaslaskurit jne. Tämän harjoitustyön aiheen valinnassa on lähinnä pidetty silmällä vakuutusalaa. Olettakaamme (kts. Beard-Pentikäinen-Pesonen [4], liite A), että riskikollektiivin vahinkojen syntymisprosessi toteuttaa seuraavat ehdot:

(i) lisäykset ovat riippumattomia
(ii) lisäykset ovat stationaarisia
(iii) samanaikaiset vahingot ovat kiellettyjä.

Tällöin todennäköisyys $p(k;t)$, että vahinkojen lukumäärä ajassa $t > 0$ on k, noudattaa Poisson-lakia

$$(2) \qquad p(k;t) = p_k(rt) = e^{-rt}\frac{(rt)^k}{k!} ,$$

jossa parametri $r > 0$ ilmoittaa vahinkojen keskimäärän aikayksikössä. On edelleen tunnettua, että välttämätön ja riittävä luonnehdinta Poisson-prosessille on, että peräkkäisten tapahtumien väliajat noudattavat eksponenttijakaumaa parametrina r. Riskiteoriassa vahinkojen lukumäärää kuvaamaan käy-

GENERATING POISSON VARIATES ON A MICROCOMPUTER

Raimo Voutilainen
Ilmarinen Pension Insurance Co. Ltd.
Eerikinkatu 41
SF-00180 Helsinki 18, Finland

The Poisson distribution is important in the mathematical considerations of both life and non-life insurance. It lends itself to e.g. birth and death processes and claim number processes. Simulation of the business flow of a company often requires generation of Poisson distributed pseudo-random numbers.

We compare the performance of two highly efficient Poisson variate generators on a TRS-80 microcomputer. The first generator is exact and the second one approximative. The analysis of the algorithms is demonstrated with help of graphical illustrations. The exact method has reliable overall performance while the approximative approach provides that the Poisson parameter is medium-sized or large. In those cases the minor programming effort required makes the approximation an attractive alternative.

tetään lähes poikkeuksetta Poisson-jakaumaa. Luonnollisesti on otettava huomioon myös yksittäisvahingon koko sekä parametrin r dynaaminen muuttuminen. Tätä varten on kehitetty yleistettyjä Poisson-jakaumia (kts. esim. Beard ym. [4], luku 3), mutta tässä työssä rajoitutaan tavalliseen Poisson-jakaumaan (1). Huomattakoon, että vahinkovakuutuksen riskiprosessien lisäksi Poisson-jakauma on hyvin merkittävä henki- ja eläkevakuutusliikkeen syntyvyys- ja kuolevuustarkasteluissa.

Viimeisten runsaan kolmen vuoden aikana vahinkovakuutusyhtiöiden solvenssi- eli vakavaraisuuskysymykset ovat olleet Suomessa tarkan analysoinnin kohteena. Sosiaali- ja terveysministeriön asettama ja professori Teivo Pentikäisen johtama solvenssityöryhmä laati laajan tietokoneohjelman vahinkovakuutusyhtiön liikkeen dynamiikan tutkimiseksi erityisesti solvenssin kannalta. Solvenssityöryhmä on koonnut työnsä kaksiosaiseksi raportiksi [10].

Tämän työn aihevalinnan taustalla olivat toisaalta em. solvenssimallin käyttöalueen laajentaminen, toisaalta riskiteorian oppikirjan [4] 3. painoksen viimeistelyn yhteydessä hahmotellut menetelmät Poisson-jakauman approksimoimiseksi. Tutkimuksen päätavoitteeksi asetettiin mahdollisimman tehokkaan algoritmin valinta joko tarkasti tai likimääräisesti Poisson-jakaumaa noudattavien satunnaislukujen generoimiseksi. Valittavan menetelmän tulee sallia jakauman (1) parametrin μ jatkuva vaihtelu, ja se sovittiin ohjelmoitavaksi TRS 80 -mikrotietokoneella, jolla ajettiin pääosa solvenssiprojektin testeistä. Yleispätevyyden ja siirrettävyyden takaamiseksi ohjelmointikieleksi valittiin EXTENDED BASIC.

Suurin osa tämän tutkimuksen tuloksista esiteltiin EURO VI -operaatiotutkimuskongressissa Wienissä heinäkuussa 1983, kts. [11].

2. Tutkittujen Poisson-generaattoreiden kuvaus

2.1. Yleistä

Generoitaessa (pseudo)satunnaislukuja muusta kuin tasajakaumasta perusaskeleena on ensin generoida satunnaisluku u tasaisesta (0,1)-jakaumasta, ja sitten etsiä luku $k = F^{-1}(u)$, jossa F on tavoitejakauman kertymäfunktio. (Kts. [4], kuva 6.8.1.) Luvut k noudattavat tällöin haluttua jakaumaa, jos luvut u ovat (riittävän tarkasti) (0,1)-tasaisesti jakautuneita. Valitettavasti kertymäfunktion käänteisfunktio on harvoin eksplisiittisesti laskettavissa, ja $F^{-1}(u)$:n laskenta vaatii hakua taulukosta, johon F-arvot on ennalta laskettu. Tämä on melko hidasta, esim. Poisson-jakauman (1) tapauksessa hakuprosessin aikavaatimus on peräkkäishakua käytettäessä $O(\mu)$, binäärihaulla $O(\log \mu)$. Sitä paitsi parametrin μ vaihtuessa taulukko on aina luotava uudestaan. Pienillä μ:n arvoilla tulevat generaattorit kuitenkin käyttävät tällaista <u>taulukkoinversiota</u>.

Keskeisestä raja-arvoväittämästä seuraa, että jos F ja N ovat Poisson- ja standardinormaalijakauman kertymäfunktioita, on suurilla μ:n arvoilla voimassa:

$$(3) \qquad F(k) \approx N((k-\mu)/\sqrt{\mu}).$$

Tätä tärkeää havaintoa ei kuitenkaan voi suoraan käyttää hyväksi Poisson-lukujen generoinnissa, ellei μ ole hyvin suuri. Normaalijakauman oikea häntä on nimittäin matalampi kuin vastaavan Poisson-jakauman, ja erityisesti vakuutussovelluksissa huomattavan suuret yksittäishavainnot ovat ratkaisevan tärkeitä.

Sitä seikkaa, että normaalijakauma on enemmän tai vähemmän tarkka approksimaatio Poisson-jakaumalle, voidaan kuitenkin käyttää, jos jotenkin korjataan virhe, joka aiheutuu Poisson-jakauman vinouden unohtamisesta kaavassa (3). Tämä on ideana molemmissa valituissa generaattoreissa, vaikka ne ovat muussa suhteessa hyvin erilaisia. Erityyppisten generointiprobleemojen palauttaminen normaalisti jakautuneiden satunnaismuuttujien

generoinniksi on yleisesti käytetty keino, koska normaalimuuttujien generointia on perin pohjin tutkittu, ja aiheesta on julkaistu lukuisia käytännöllisiä algoritmeja, kts. esim. Ahrens-Dieter [1] ja Knuth [8].

2.2. Ahrensin ja Dieterin algoritmi

Valituista generointialgoritmeista ensimmäinen on Ahrensin ja Dieterin menetelmä [3]. Tämän ja eräiden aikaisempien Ahrensin ja Dieterin tutkimuksien lähtökohtana on seuraava D. Knuthin esittämä tutkimustehtävä ([8], luku 3.4.1, harjoitustehtävä 22): "Can the exact Poisson distribution for large μ be obtained by generating an appropriate normal deviate, converting it ot an integer in some convenient way, and applying a (possibly complicated) correction a small percent of the time?"

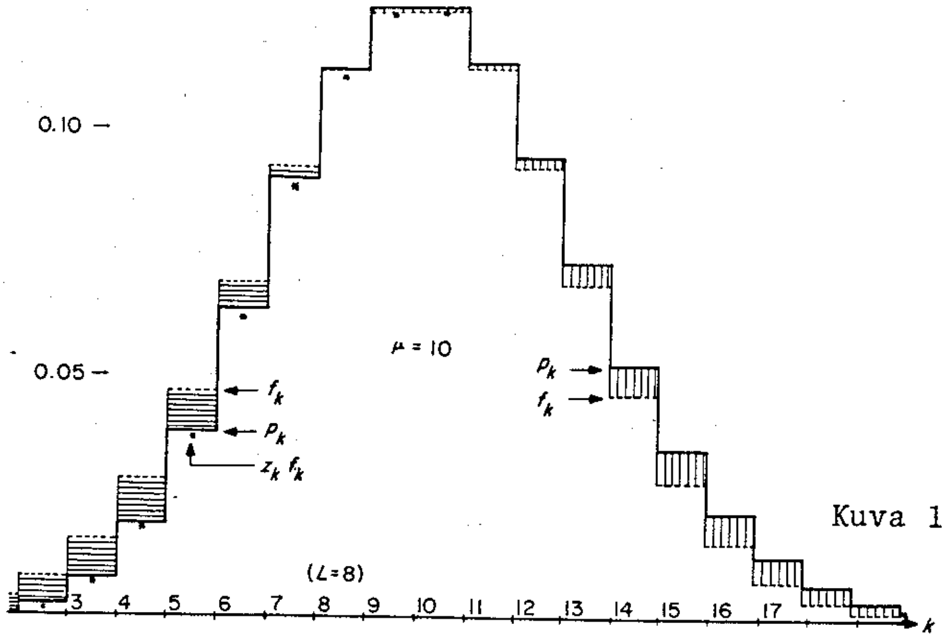
Sekä diskreettien että jatkuvien satunnaismuuttujien generoinnissa sangen yleinen menetelmä on "hyväksy-hylkää -tekniikka" (acceptance-rejection technique). Kohdejakauman asemasta siinä muodostetaan satunnaisluku jonkin helposti laskettavan majoranttitiheysfunktion määrittelemästä jakaumasta, ja tämä luku hyväksytään todennäköisyydellä, joka yhtyy tavoitejakauman tiheysfunktion ja sitä ylhäältä rajoittavan apufunktion arvojen osamäärään tässä pisteessä. Hyväksyminen tai hylkääminen ratkaistaan käyttämällä tasajakaumageneraattoria. Majoranttifunktio, jota usein myös kutsutaan "hatuksi", määritellään tavallisesti paloittain siten, että vaikeasti laskettavien jakauman häntien hattuna on esim. yksinkertainen eksponenttifunktio. Menetelmän kriittinen kohta on em. osamäärän laskeminen, jossa usein joudutaan turvautumaan vaikeahkoihin sarjakehitelmiin. Tuloksena olevan generointialgoritmin tehokkuus riippuu ratkaisevasti myös siitä, kuinka tarkasti hattufunktio approksimoi tavoitejakaumaa: esim. yksinkertaisten porrasfunktioiden käyttö johtaa tavallisesti liian lukuisiin hylkäyksiin ja aikaa vieviin tasajakaumageneraattorin kutsuihin.

Yllä mainitussa lainauksessa Knuthin oppikirjasta [8] viitataan mahdollisuuteen käyttää normaalijakaumaa Poisson-jakauman hattuna. On kuitenkin helppo nähdä, että Poisson-jakauman oikea häntä ei mahdu minkään normaalijakauman tiheysfunktion

alapuolelle. Artikkelissa [2] Ahrens ja Dieter käyttivät hattuna kaksiosaista eksponenttifunktiota. Julkaistu algoritmi oli erittäin merkittävä, koska se oli ensimmäinen eksakti Poisson-generaattori, jonka aikavaatimuksella on täysin μ:stä riippumaton yläraja. Ainoa menetelmän tarvitsema apugeneraattori on (0,1)-tasajakaumageneraattori. Numeerisesti pahanlaatuisista laskutoimituksista selviämiseksi algoritmissa käytetään kaksoistarkkuuden aritmetiikkaa, joka todennäköisesti heikentää sen käytettävyyttä mikrotietokoneessa.

Vuonna 1982 Ahrens ja Dieter julkaisivat edellistä tehokkaamman algoritmin [3], joka on ilmeisesti tällä hetkellä nopein eksakti Poisson-generaattori. Algoritmi sallii jatkuvan μ-parametrin vaihtelun. Sen FORTRAN-versio Siemens 7760 -tietokoneella kuluttaa yhden Poisson-luvun generoimiseen keskimäärin 100-150 mikrosekuntia riippuen siitä, onko μ kiinteä vai vaihdellaanko sitä toistuvasti. Tämä edellyttää, että algoritmin kutsumat apugeneraattorit ovat saatavissa konekielisinä. Jos koko algoritmi ohjelmoidaan konekielellä, generointiaika putoaa 50-90 mikrosekuntiin, joka on jo varsin lähellä yksinkertaisen tarkkuuden logaritmin suoritusaikaa (esim. Siemens 7760:llä 50 mikrosekuntia). Näin ollen voitaneen olettaa, että eksaktin Poisson-generoinnin tehostaminen tästä ei enää ainakaan oleellisessa määrin ole mahdollista, ja artikkelin [3] algoritmi valittiinkin tässä työssä ohjelmoitavaksi ja kokeiltavaksi.

Ahrensin ja Dieterin algoritmi approksimoi Poisson-jakaumaa sellaisella diskretisoidulla normaalijakaumalla, että vasemman hännän ylimääräisellä todennäköisyysmassalla on sama pinta-ala kuin oikean hännän "vajauksella". Oheisessa kuvassa 1, jossa $\mu = 10$, Poisson- ja normaalijakaumien pistetodennäköisyysfunktioita merkitään p_k:lla ja f_k:lla. (Luvut f_k ovat standardinormaalijakauman tiheysfunktion integraaleja vastaavien välien yli.) Käytettävä "diskreetti normaalijakauma" ei selvästikään ole Poisson-jakauman hattu hyväksy-hylkää -tekniikan tarkoittamassa mielessä. Se ei edes ole tasaisesti paras Poisson-jakauman normaaliapproksimaatio. Se on huolellisten kokeiden jälkeen määritelty siten, että $p_k < f_k$ kaikilla $k < m$ ja $p_k \geq f_k$ kaikilla $k \geq m$, missä $m \leq L = \lfloor \mu - 1.1484 \rfloor$, jos $\mu \geq 10$. ($\lfloor\ \rfloor$ on lattiafunktio eli argumenttinsa kokonaisosa.)



Kuva 1.

Huomattavasti yksinkertaistettuna Ahrensin ja Dieterin algoritmi on seuraava:

<u>Algoritmi PD</u> (input: parametri μ, output: Poisson(μ)-jakautunut satunnaisluku K)

PD1. Jos $\mu \geq 10$, mene askeleeseen PD2. Muussa tapauksessa laske K taulukkoinversiolla ja lopeta. (Jos μ on sama kuin edellisellä kutsukerralla, ei Poisson-kertymäfunktion arvoja tarvitse laskea uudelleen.)

PD2. Generoi satunnaisluku T N(0;1)-jakaumasta ja aseta $K \leftarrow \lfloor \mu + \sqrt{\mu} T \rfloor$. Jos $K \geq L$ (kts. ed.) hyväksy K ja lopeta ($p_K \geq f_K$).

PD3. ($p_K < f_K$.) Hyväksy-hylkää -testi: Generoi U (0,1)-tasajakaumasta. Jos $U \leq p_K / f_K$, hyväksy K ja lopeta. (Varsinkin suurilla μ:n arvoilla p_K:n ja f_K:n laskenta on työlästä ja vaatii mm. Stirlingin approksimaatiota ja Hermiten polynomeja. U:ta voidaan kuitenkin verrata yksinkertaiseen funktioon $z_K \leq p_K / f_K$, jolloin p_K:n ja f_K:n laskenta on tarpeen vain, kun $U > z_K$. z_K on p_K / f_K:n tarkka ala-arvio; kuvan 1 tähdet osoittavat $z_K f_K$:n olevan lähellä p_K:ta.)

matta hyvin nopeat laskutoimitukset riittävät yli 90 %:ssa tapauksia. Kun $\mu \geq 10$, algoritmi kutsuu normaalijakaumageneraattoria askeleessa PD2, tasajakaumageneraattoria askeleissa PD3 ja PD4 ja standardieksponenttijakaumageneraattoria (parametri 1) askeleessa PD4. Tarkemmat yksityiskohdat käyvät ilmi artikkelista [3].

2.3. Pentikäisen ym. approksimaatioalgoritmi

Toiseksi Poisson-generaattoriksi on valittu approksimaatiomenetelmä, joka on hahmoteltu Pentikäisen ym. riskiteorian oppikirjan tulevassa 3. painoksessa [4]. Normaalijakauma on tyypillinen kaksiparametrinen jakauma, jota voidaan käyttää annetun jakauman approksimoimiseen ja jonka keskiarvo ja keskihajonta voidaan sovittaa täsmäämään alkuperäisen jakauman vastaaviin momentteihin. Tämä toteutetaan usein sovittamalla standardi N(0;1)-jakauma normeerattuun kohdejakaumaan. Mitä vinompi alkuperäinen jakauma on, sitä heikompi on approksimaatio. Vinous voidaan ottaa huomioon käyttämällä epätäydellisen gammafunktion kolmiparametrista versiota ([4], 3.5.6, s. 70). Olkoon $\underset{\sim}{Z}$ mielivaltainen satunnaismuuttuja ja olkoon $\underset{\sim}{z} = (\underset{\sim}{Z} - \mu)/\sigma$ vastaava normeerattu muuttuja. Jos Z ja z ovat muuttujien $\underset{\sim}{Z}$ ja $\underset{\sim}{z}$ vastinarvoja ja jos γ on $\underset{\sim}{Z}$:n (ja $\underset{\sim}{z}$:n) vinous, voidaan osoittaa, että $\underset{\sim}{Z}$:a approksimoiva kolmiparametrinen gammafunktio on muotoa

$$\text{(4)} \qquad \bar{S}(z) = S(Z) = \Gamma(\alpha + z\sqrt{\alpha}\,,\, \alpha) = \frac{1}{\Gamma(\alpha)} \int_0^{\alpha + z\sqrt{\alpha}} e^{-u}\, u^{\alpha-1}\, du \qquad (z \geq -\sqrt{\alpha}),$$

jossa $\alpha = 4/\gamma^2$. Kertymäfunktion (4) määrittelemällä jakaumalla on sama odotusarvo (=0), keskihajonta (=1) ja vinous γ kuin $\underset{\sim}{z}$:lla.

Gammafunktion (4) numeerista laskentaa varten oppikirjan [4] tekijät suosittelevat potenssisarjakehitelmää, jos γ on verrattain suuri. Pieniä γ:n arvoja varten he ehdottavat seuraavaa Wilson-Hilfertyn approksimaatiokaavaa (kts. [7], luku 17.5):

(5) $$S(Z) \approx N(c_1 + c_2(z + c_3)^{1/3}),$$

jossa

(6) $$c_1 = \gamma/6 - 6/\gamma, \quad c_2 = 3 \cdot (2/\gamma)^{2/3}, \quad c_3 = 2/\gamma.$$

Näitä tuloksia sovelletaan nyt Poisson(μ)-satunnaismuuttujaan k. Normeeraamalla saadaan

(7) $$z = (k - \mu + 0.5)/\sqrt{\mu}.$$

Erona kaavan (7) ja yhteyskaavan (3) s. 3 välillä on jatkuvuuskorjaus 0.5, jonka voidaan odottaa nopeuttavan z:n jakauman suppenemista kohti N(0;1)-jakaumaa, kun μ kasvaa. Samanlaisen korjauksen tekee mm. Rubinstein [9], kaava 3.7.28 s. 103. Ottamalla vielä huomioon, että z:n (ja k:n) vinous on $1/\sqrt{\mu}$, saadaan algoritmi Poisson(μ)-muuttujan generoimiseksi: Aluksi generoidaan satunnaisluku y N(0;1)-jakaumasta. Kaavan (5) mukaisesti kirjoitetaan

(8) $$y = c_1 + c_2(z + c_3)^{1/3}.$$

Sijoittamalla yhtälöön (8) termit (6) ja (7) ja ratkaisemalla k näin saadusta yhtälöstä tuloksena on

(9) $$k = \frac{1}{108\sqrt{\mu}}\left(y + 6\sqrt{\mu} - \frac{1}{6\sqrt{\mu}}\right)^3 - \mu - 0.5.$$

Jos lauseke (9) on negatiivinen, k hylätään, uusi satunnaisluku y generoidaan ja rivin (9) laskutoimitus toistetaan. (Lauseke (9) voi käytännössä olla negatiivinen vain pienillä μ:n arvoilla, joilla menetelmää ei ole tarkoitettu sovellettavaksi.) Muussa tapauksessa hyväksytään $\lfloor k \rfloor$ Poisson(μ)-jakautuneeksi satunnaisluvuksi. (Vaihtoehtoisesti k voitaisiin pyöristää lähimpään kokonaislukuun = $\lfloor k + 0.5 \rfloor$, mutta samaan tapaan kuin esim. Rubinsteinin [9] kaavassa 3.7.29 s. 103 tässä on valittu $\lfloor k \rfloor$. Kyseeseen tulevalla μ-alueella tällä seikalla ei ole sanottavaa merkitystä.) Korostettakoon vielä, että ollakseen tarkka menetelmä vaatii melko pienen vinouden γ, ts. μ:n on oltava melko

suuri. Tämän tutkimuksen keskeisiä tehtäviä oli kokeellisesti selvittää alhaisimmat kyseeseen tulevat μ:n arvot.

2.4. Apugeneraattorit

Koska molemmat edellä kuvatut Poisson-satunnaislukugeneraattorit tarvitsevat aliohjelmakseen N(0;1)-generaattorin, oli ensin löydettävä tehokas normaalijakaumageneraattori, joka soveltuisi lausekieliseen ohjelmointiin. Boxin ja Mullerin tunnettu sini-kosini -menetelmä [5] oli osoittautunut käyttökelpoiseksi solvenssiprojektissa. Toinen tässä työssä testattu tehokas ja eksakti algoritmi on Ahrensin puolisuunnikasmenetelmä, kts. [1]. Koska Boxin ja Mullerin menetelmä on huomattavasti yksinkertaisempi ja TRS-80:n BASIC-ympäristössä vain 20-30 % hitaampi, se valittiin Poisson-generaattorien aliohjelmaksi.

Ensimmäinen Poisson-generaattori tarvitsee myös satunnaislukuja standardieksponenttijakaumasta. Tämä on yksi harvoista standardijakaumista, joille voidaan laatia satunnaislukugeneraattori kääntämällä eksplisiittisesti kertymäfunktio. Generaattori on äärimmäisen yksinkertainen: Generoi ensin satunnaisluku u (0,1)-tasajakaumasta, sitten laske ja tulosta $x = -\ln u$. Korkean tason ohjelmointikieltä käytettäessä tämä menetelmä on nopein mahdollinen (vrt. Ahrens-Dieter [1], s. 876), joten se valittiin tähän tarkoitukseen. (0,1)-tasajakaumageneraattorina käytettiin TRS-80:n LEVEL II BASIC -järjestelmän systeemifunktiota RND(0).

3. Generaattoreiden tietokonetoteutus

3.1. Käytetty tietokonelaitteisto

Luvussa 2. kuvatut Poisson-generaattorit ohjelmoitiin ja testattiin TRS-80 model I -mikrotietokoneella, johon on kytketty laajennusliitäntä, kirjoitin, piirturi ja kaksi minidiskettejä käyttävää levyasemaa. Sanan pituus on 8 bittiä, ja keskus-

muistia on 48 kilosanaa. Prosessorin yksinkertainen tarkkuus on 7 desimaalia (kaksoistarkkuus on 17 desimaalia, mutta sitä ei tässä sovelluksessa tarvinnut käyttää). Käytössä oli EXTENDED BASIC -tulkki, vaikka ohjelmien nopeuttamiseksi olisi mahdollista käyttää myös BASIC-kääntäjää. Generaattoreiden yhteyteen laadittiin ohjelman osat, jotka mahdollistavat yhteyden prof. Pentikäisen matriisinpiirto-ohjelmiin. Näitä ohjelmia käytettiin luvussa 3.4. ja liitteessä 1 olevien graafisten esitysten valmistamiseen.

3.2. Ohjelman käyttö

Molemmat generaattorit sisältävän BASIC-ohjelman listaus on liitteenä 2. Ohjelman rakenteesta todettakoon vain, että alkudialogin ja määrittelyjen jälkeen seuraa Ahrensin ja Dieterin algoritmi riveillä 3240-3600, ja Pentikäisen ym. algoritmi riveillä 3680-3780. Viimeksi generoitu satunnaisluku on muuttujan K arvona, ja se talletetaan tarpeen mukaan tulostuspuskuriin (rivit 3850-3880) ja frekvenssitaulukkoon levylle viemistä varten (rivit 3890-3980).

Ohjelman toimintaa selventää erään esimerkkiajon syöttö-tulostus -dialogi liitteessä 3. Alkutiedonantojen jälkeen kone kysyy, kuinka monta Poisson-lukua ensimmäisellä μ:n arvolla generoidaan. Esimerkkiajossa käyttäjä haluaa tulostaa satunnaisluvut sekä koota niistä graafisen esityksen, joista esimerkkejä on liitteessä 1. Tämän jälkeen annetaan ensimmäinen Poisson-parametri, ja valitaan jompi kumpi kahdesta generaattorista. Ensimmäinen valinta on Ahrensin ja Dieterin menetelmä, ja suorituksen alussa ja lopussa kone ilmoittaa generoitavien lukujen määrän, μ-parametrin sekä päiväyksen kellonaikoineen. Nämä kontrollirivit tulostuvat myös kirjoittimelle. Niiden välissä on 200 Poisson(50)-jakautunutta satunnaislukua 10:n erissä, kuten liitteen 3 jälkimmäiseltä sivulta nähdään. Koska luvuista halutaan graafinen esitys, kone kysyy, millä nimellä laskettu frekvenssivektori halutaan tallettaa. Jos levytalletus onnistuu (disketillä on tilaa), generointia voidaan välittömästi jatkaa. Tällä kertaa käyttäjä haluaa 100 Poisson(50)-jakautunutta satunnaislukua, jotka halutaan kirjoittimelle, mutta joita ei välitetä tallettaa

frekvenssitaulukkoon. Tällä kertaa käytetään Pentikäisen ym. approksimaatiomenetelmää. Jos parametri on pieni, voidaan tässä vaiheessa käyttäjän harkinnan mukaan siirtyä taulukkoinversioon. (Tämä ohjelman osa on yhteinen molemmille generaattoreille.) Generaattorin saatua työnsä valmiiksi ohjelma kysyy, halutaanko generointia edelleen jatkaa. Tässä tapauksessa ei haluta, ja nyt on tilaisuus tallettaa levylle korrektit Poisson-todennäköisyydet simuloinnin tulosten arvioimiseksi (vrt. porrasfunktiot liitteessä 1). Tässä vaiheessakin voidaan vielä palata generoimaan uusia satunnaislukuja. Normaalissa tuotantokäytössä testiajojen ulkopuolella simulointitulosten graafisella tarkastelulla ei tietenkään yleensä ole merkitystä, koska menetelmien tarkkuus on tässä työssä tyhjentävästi selvitetty, kts. luku 3.4.

3.3. Muistitila- ja aikavaatimukset

Tarkasteltava BASIC-ohjelma mahtui 2.5:lle minidisketin uralle, joten sen tallettaminen vaati n. 6.25 kilotavua. (Disketin kapasiteetti normaalikäytössä on 87.5 kilotavua.) Niinpä ohjelma vie keskusmuistissakin vain murto-osan käyttäjälle varatusta muistialueesta.

Molemmilla testatuilla menetelmillä on se erinomainen ominaisuus, että yhtä satunnaislukua varten käytettävä aika ei kasva μ:n kasvaessa. Tilanne ei kummassakaan algoritmissa oleellisesti muutu, vaikka μ:tä vaihdeltaisiin jatkuvasti. Luonnollisesti aikavaatimus on vaadittujen Poisson-lukujen lukumäärän lineaarinen funktio. Menetelmien ajankäytössä ei ollut merkittävää eroa millään μ:n arvoilla. Kun $\mu \geq 10$, syntyi vain muutaman prosentin eroja. Ilmeisesti menetelmä 1 onnistuu tehokkaasti välttämään suurimmassa osassa tapauksista askelten PD3 ja PD4 hankalat laskutoimitukset, kts. luku 2.2., ss. 6-7. Vaikka samalla ohjelmakoodilla voitaisiin nykyaikaisella suurtietokoneella generoida tuhansia satunnaislukuja sekunnissa, saatiin nyt vain 3-4 Poisson-lukua sekunnissa. Henkilökohtaisessa tietojenkäsittelyssä tämä lienee kuitenkin riittävää useimpiin sovellutuksiin.

3.4. Tarkkuusarvioita

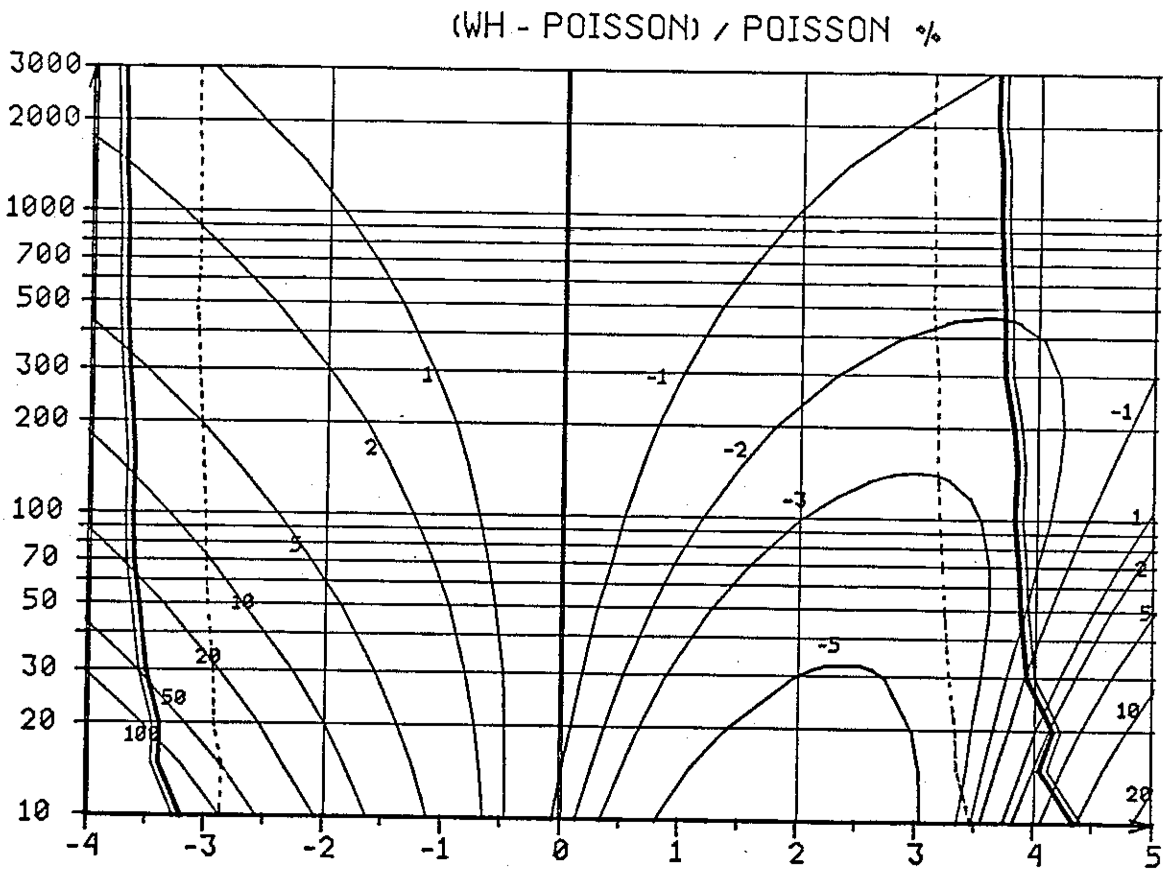
Menetelmien tarkkuutta testattiin graafisesti. Koska menetelmä 1 on eksakti, menetelmävirhettä ei ole, ja tarkkailtavaksi jää simulointivirhe. Riittää siis tutkia, kuinka monta satunnaislukua on generoitava, jotta niiden jakauma olisi riittävän lähellä tarkkaa Poisson-jakaumaa. Liitteen 1 kuvissa tarkat Poisson-todennäköisyydet on piirretty katkopylväinä pienillä μ:n arvoilla, ja porrasfunktiona keskisuurilla μ:n arvoilla. Tällöin porraskohdat ovat vastaavien pylväiden keskikohdassa, ja pylvästä vastaa oikeanpuoleinen porras. Näin menetellen kuviot on saatu paremmin erottumaan toisistaan.

Kun $\mu < 10$, menetelmä 1 käyttää taulukkoinversiota. Esimerkkitulokset 100, 300, 1 000 ja 10 000 luvun generoinnista jakaumasta Poisson(5) ovat liitteen 1 sivulla 1. Simulointivirhettä ja tarvittavien lukujen määrää arvioitaessa tulee varmistua erityisesti siitä, että oikeanpuoleinen häntä on suunnilleen oikean muotoinen. 3 000 generointia antaa jo varsin luotettavan kuvan jakaumasta. Jakauman keskellä saattaa vielä esiintyä satunnaisia piikkejä, vrt. liitteen 1 sivu 2 ($\mu = 30$). Sivu 3 antaa kuvaa suppenemisesta suuremmilla μ:n arvoilla (tässä $\mu = 1\ 000$).

Menetelmä 2 on likimääräinen, ja sen tarkkuuden voi odottaa kasvavan μ:n mukana. Kuten luvussa 2.3. todettiin, sitä ei ole lainkaan tarkoitettu hyvin pienille μ:n arvoille. Niissä tapauksissa lauseke (9) s. 9 on useammin negatiivinen, ja hylkäykset lisäävät ajankulutusta. Lausekkeen (9) määrittelemässä Wilson-Hilfertyn jakaumassa on vasemmalla hännällä suuri ylijäämä, joka vähenee μ:n kasvaessa, kts. liitteen 1 sivu 4. Sivu 5 osoittaa konvergenssiä μ:n arvolla 50. Wilson-Hilfertyn jakaumassa on vielä selvä vääristymä, mutta oikea häntä alkaa olla lähellä oikeaa. μ:n arvolla 85 s. 6 approksimaatio edelleen paranee. Suuremmilla μ:n arvoilla menetelmien 1 ja 2 väliset erot vähitellen häipyvät, kts. $\mu = 1\ 000$ s. 3 ja 7.

Tämän työn päätehtävä on menetelmän 2 käyttökelpoisuuden arvioiminen. Häiritsevä piirre liitteen 1 kuvissa on se, että niissä simulointi- ja menetelmävirheet vaikuttavat päällekkäin, eikä niitä varsinkaan jakauman kriittisen oikean hännän osalta ole mahdollista erottaa toisistaan. Jotta simulointivirhe saatai-

siin tarkastelusta täydellisesti eliminoiduksi, on kuvassa 3 laskettu Wilson-Hilfertyn jakauman tiheysfunktion suhteellinen virhe verrattuna vastaavan Poisson-jakauman pistetodennäköisyysfunktioon.



Kuva 3

Pystyakseli on Poisson-parametri μ, ja vaaka-akseli kuvaa satunnaismuuttujien saaman arvon k etäisyyttä odotusarvosta hajonnan monikertoina, ts. suuretta $(k - \mu)/\sqrt{\mu}$. Kuvioon on piirretty tärkeimmät kuvattavan kahden muuttujan funktion tasa-arvokäyrät maastokartan tapaan. Paksun ja ohuen yhtenäisen käyrän muodostamat putket rajaavat ulkopuolelleen alueen, jossa Poisson-jakauman vasemman/oikean hännän todennäköisyysmassa on alle 10^{-4}. Katkoviivavyöhykkeen ulkopuolella häntien massat ovat alle 10^{-3}.

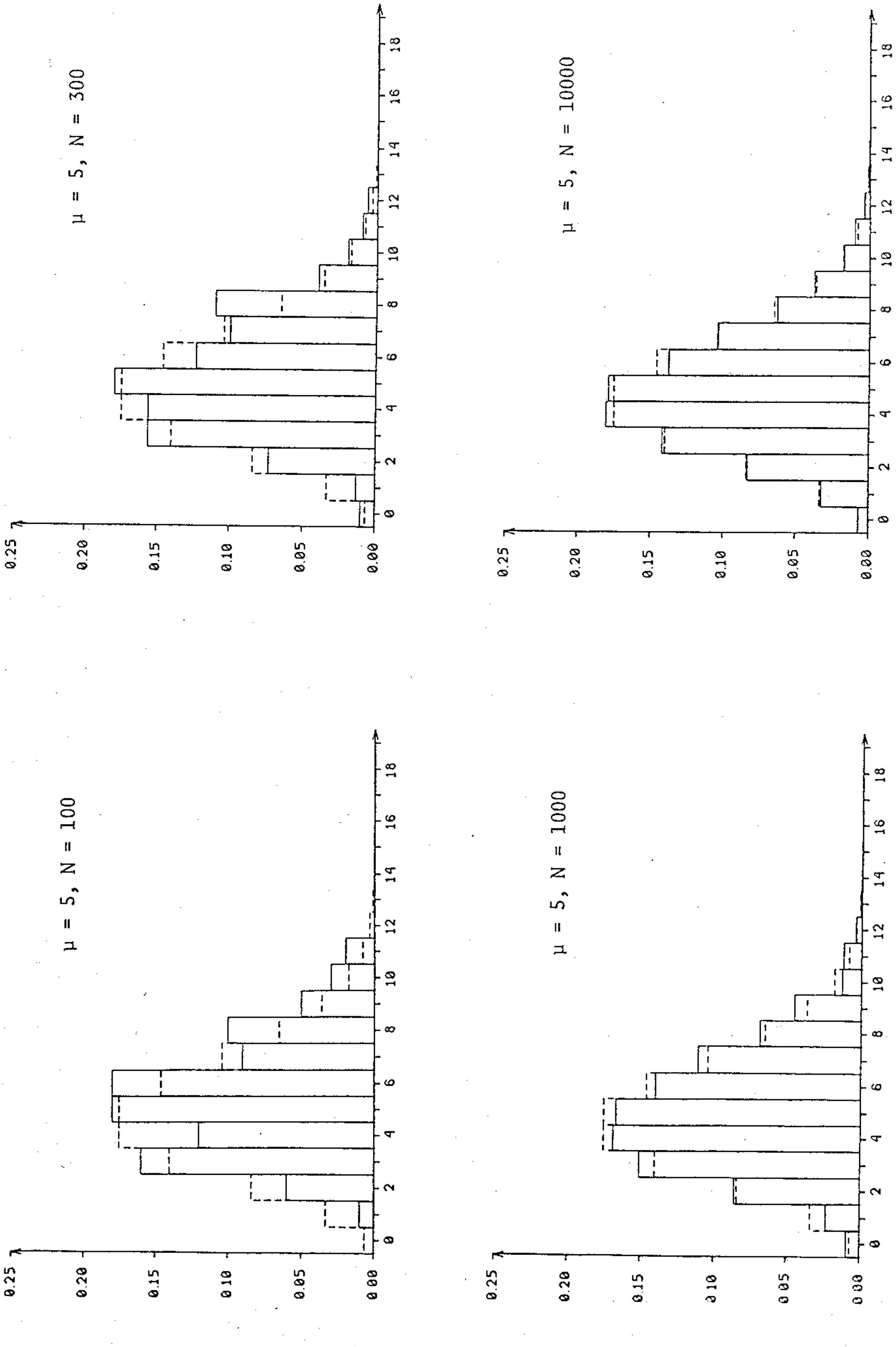
Kuvan 3 kartan mukaan on helppo päätellä, onko Wilson-Hilfertyn approksimaatiota turvallista käyttää. Sovelluskohteesta ja tarkkuusvaatimuksesta riippuen hyväksymisraja lienee jossakin μ:n arvon 100 tienoilla.

4. Johtopäätöksiä

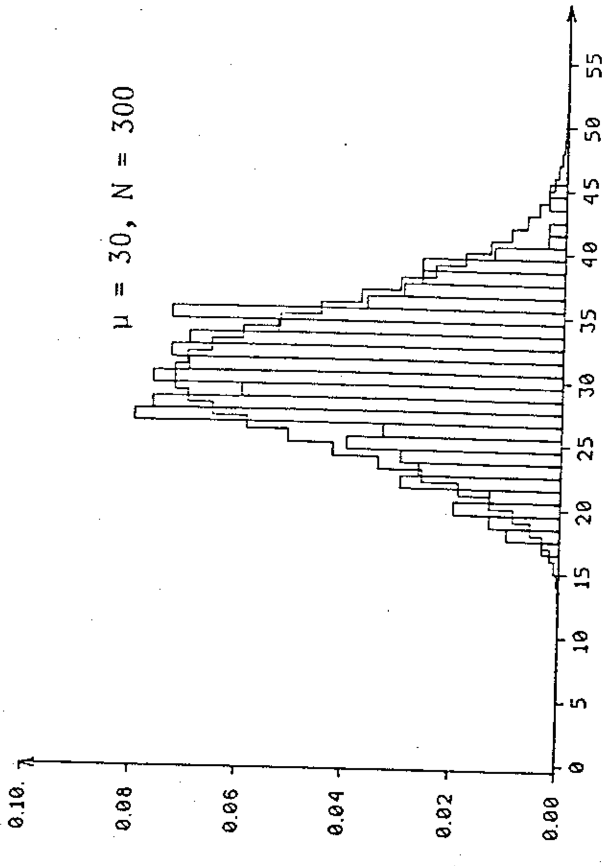
Kuten luvun 3 tietokoneanalyysistä käy ilmi, eksaktia menetelmää 1 tulisi käyttää, n. μ-arvon 70 alapuolella. Jos molemmat menetelmät on tietokoneeseen ohjelmoitu, voidaan menetelmää 1 tietysti suositella täysin turvallisena myös muille parametrin arvoille. Kuten kuvaa 3 tarkasteltaessa todettiin, menetelmä 2 on jokseenkin turvallinen parametrialueella $\mu \geq 100$. Se on huomattavasti helpompi ohjelmoida, joka tekee siitä suositeltavan ainakin satunnaisessa käytössä.

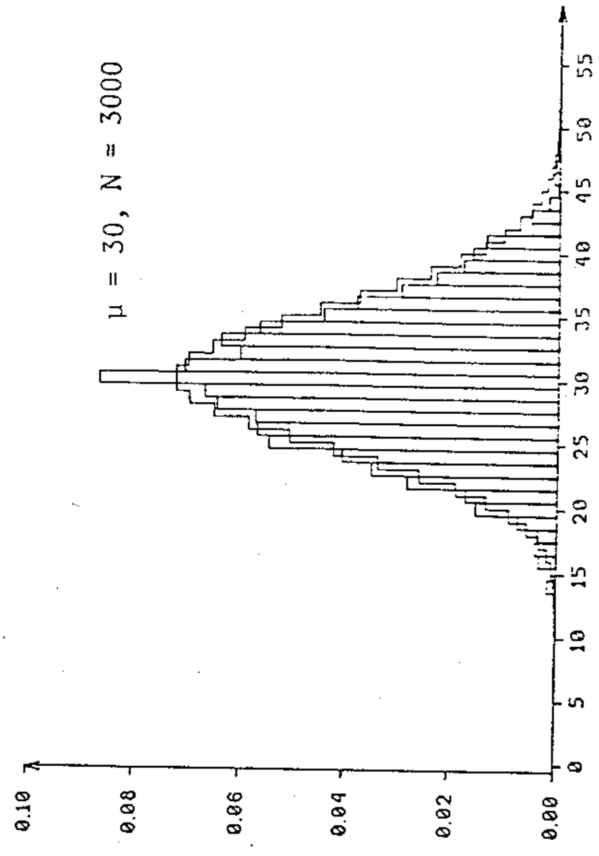
Haluan esittää parhaat kiitokseni tämän työn ohjaajalle prof. Teivo Pentikäiselle kannustuksesta, mielenkiintoisista keskusteluista ja arvokkaista huomautuksista.

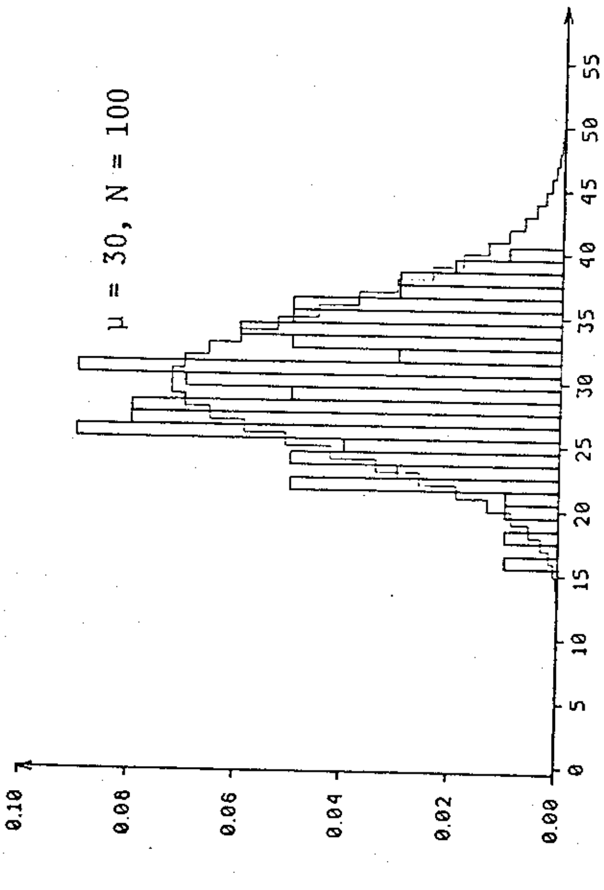
Menetelmä 1

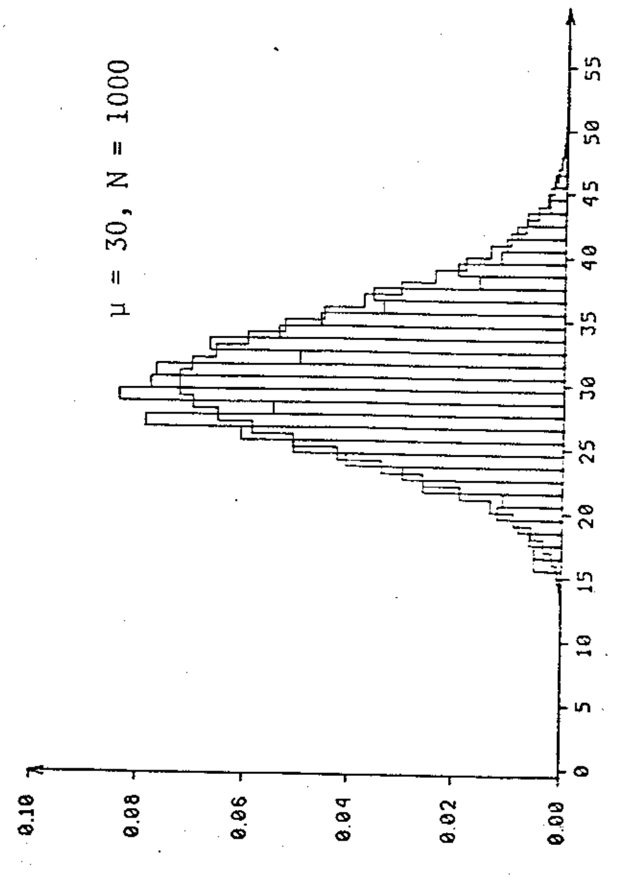


Menetelmä 1

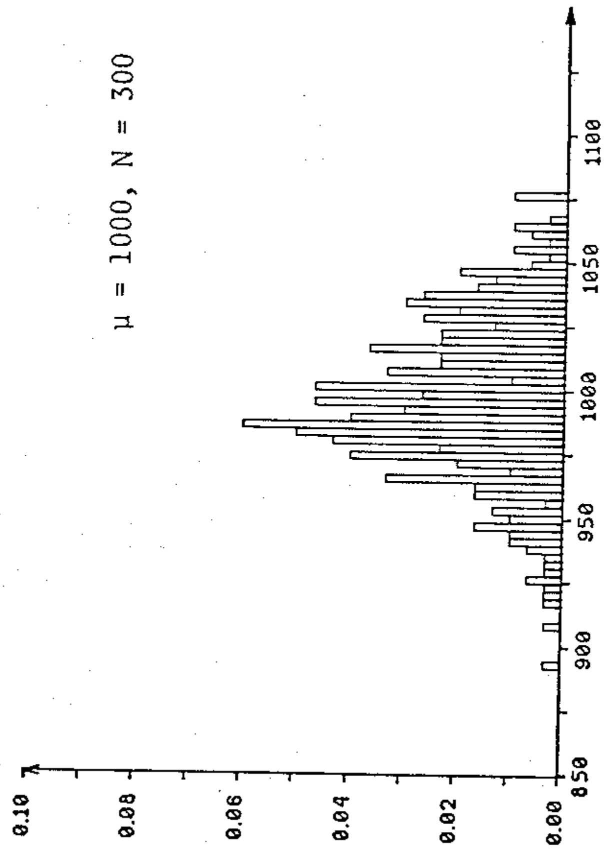


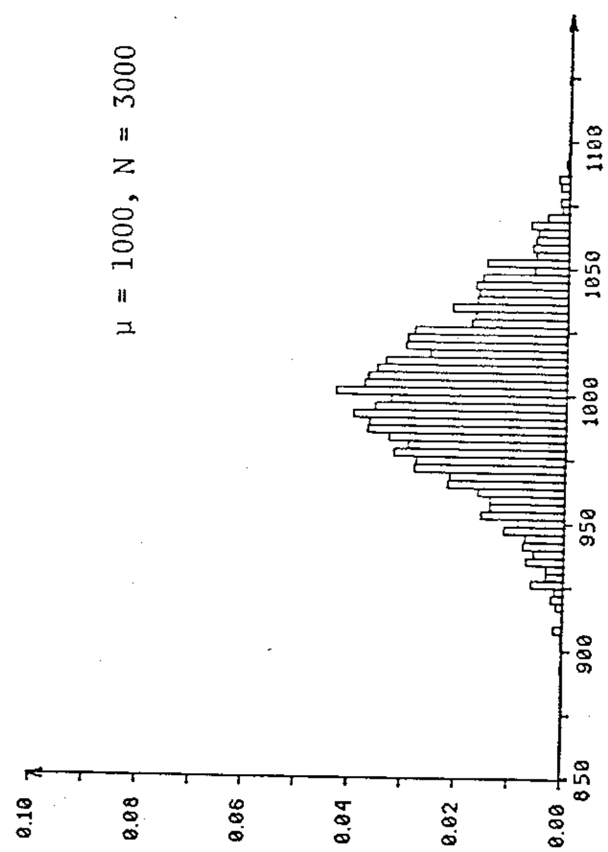


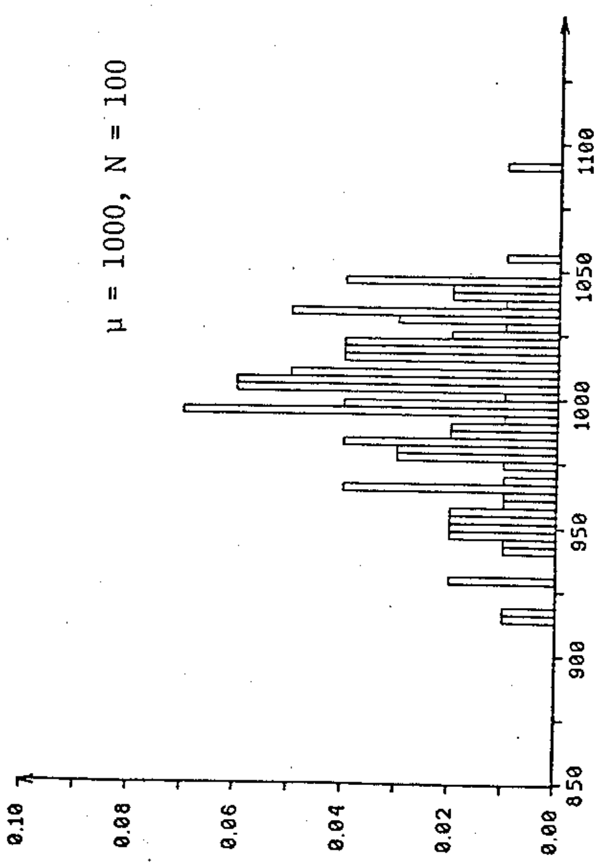


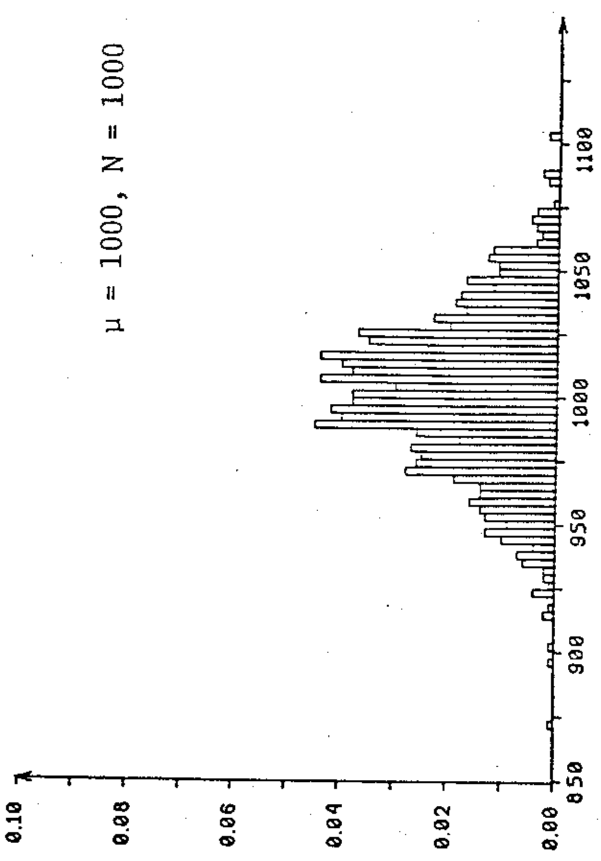


Menetelmä 1

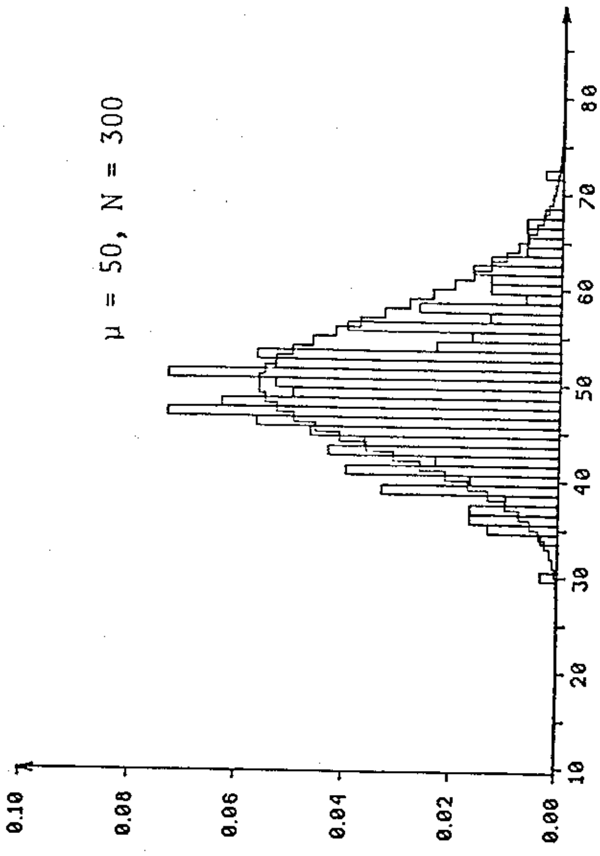

μ = 1000, N = 300
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100

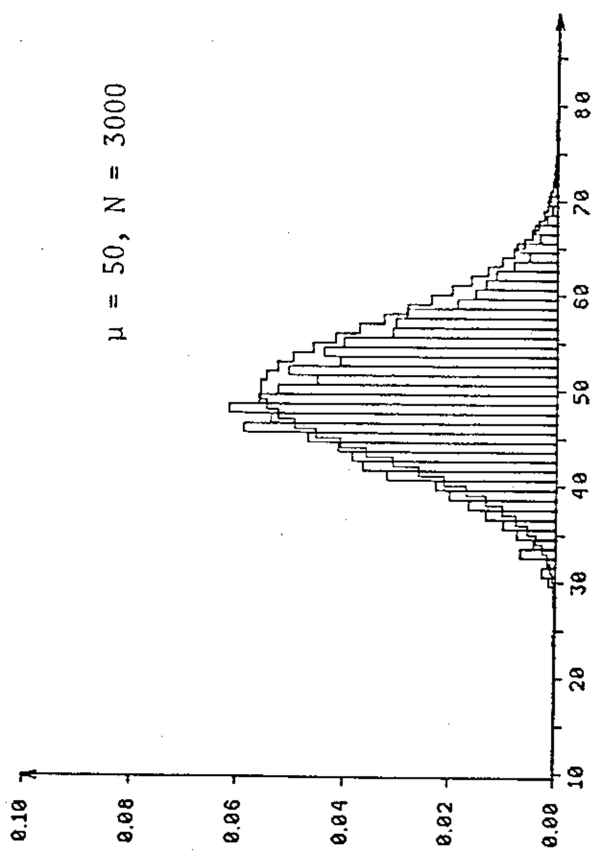

μ = 1000, N = 3000
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100

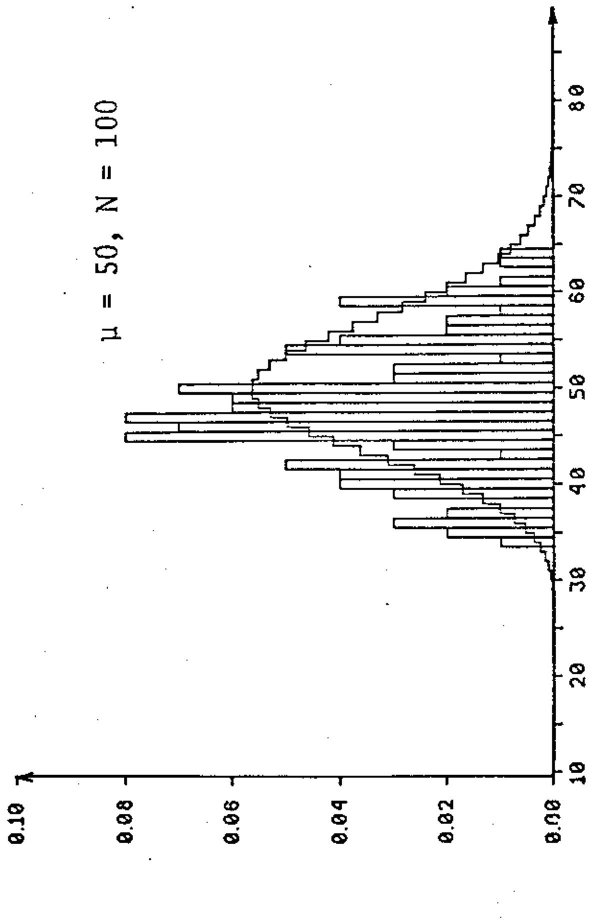

μ = 1000, N = 100
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100

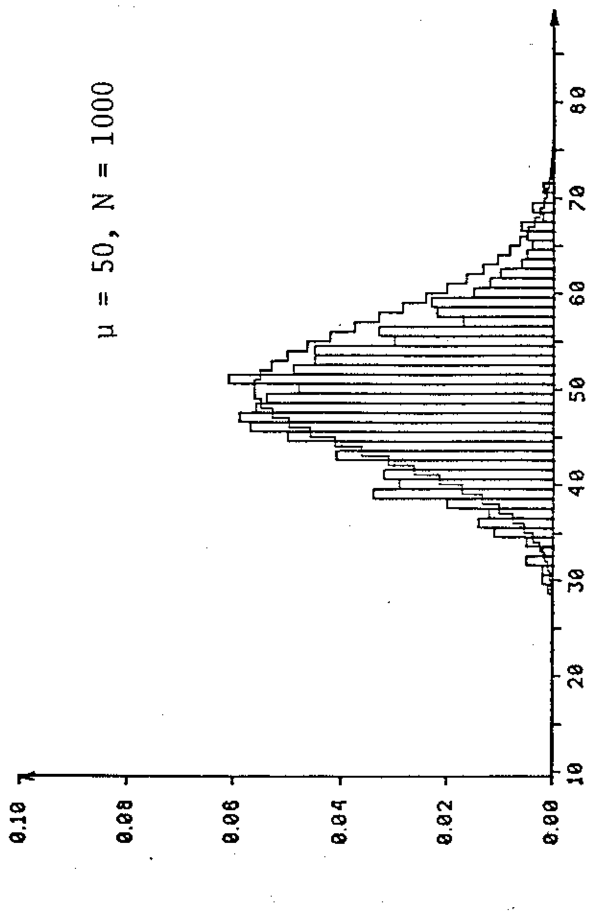

μ = 1000, N = 1000
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100

Menetelmä 2

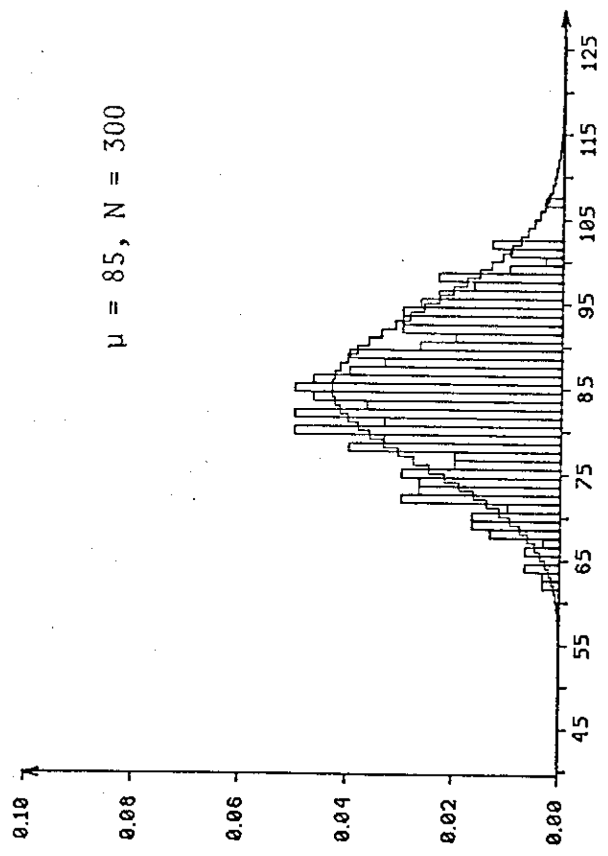

μ = 50, N = 300
0.10
0.08
0.06
0.04
0.02
0.00
10
20
30
40
50
60
70
80

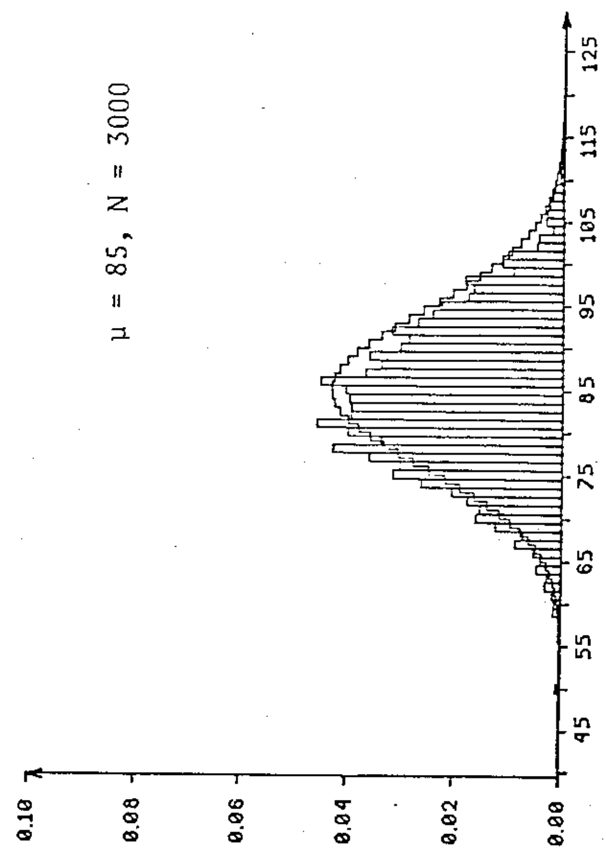

μ = 50, N = 3000
0.10
0.08
0.06
0.04
0.02
0.00
10
20
30
40
50
60
70
80

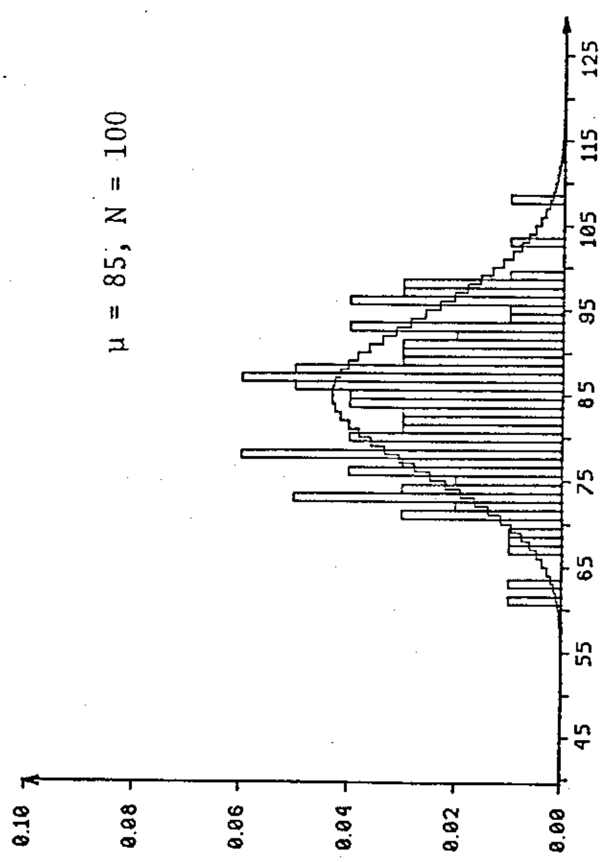

μ = 50, N = 100
0.10
0.08
0.06
0.04
0.02
0.00
10
20
30
40
50
60
70
80

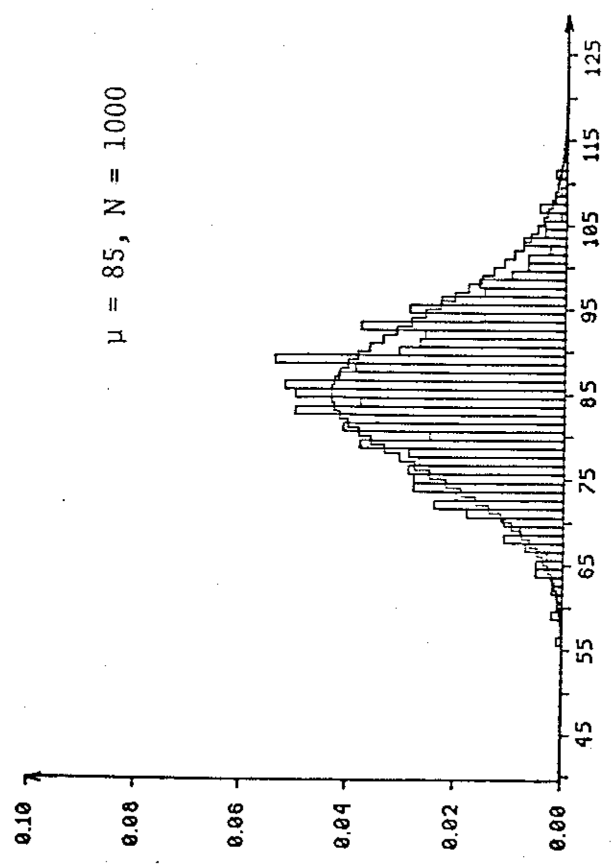

μ = 50, N = 1000
0.10
0.08
0.06
0.04
0.02
0.00
10
20
30
40
50
60
70
80

Menetelmä 2

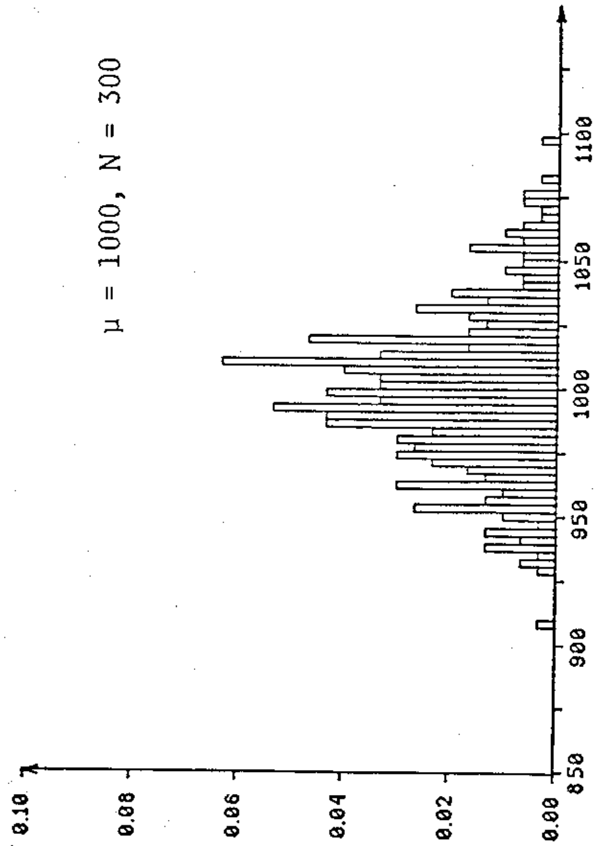

μ = 85, N = 300
0.10
0.08
0.06
0.04
0.02
0.00
45
55
65
75
85
95
105
115
125

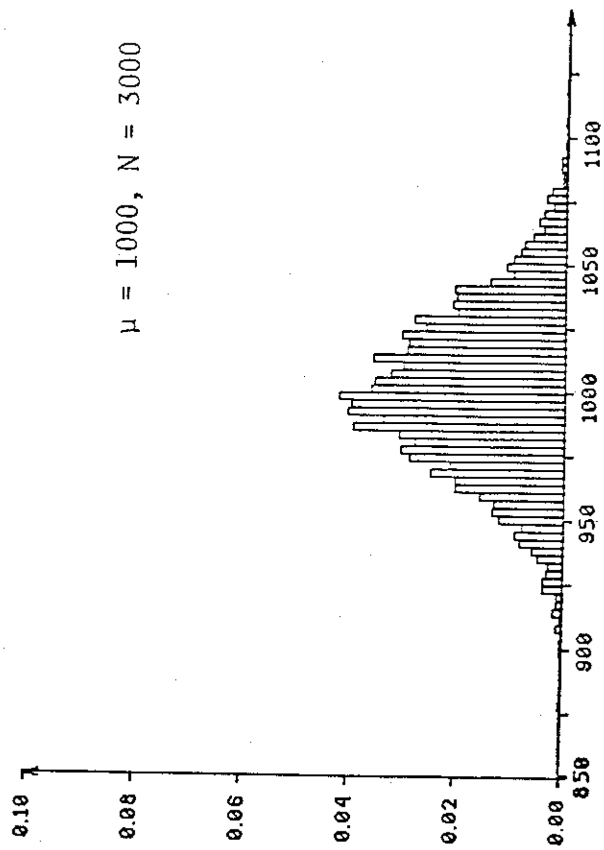

μ = 85, N = 3000
0.10
0.08
0.06
0.04
0.02
0.00
45
55
65
75
85
95
105
115
125

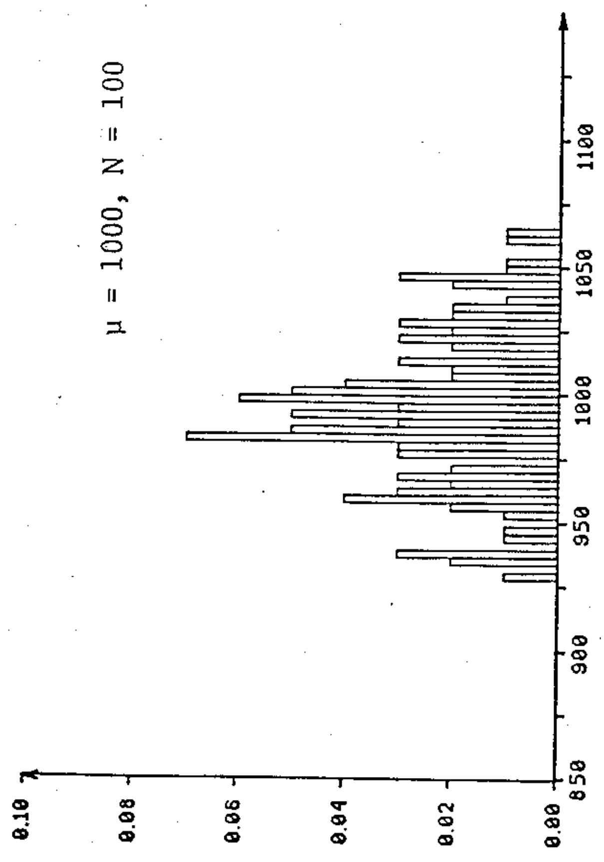

μ = 85, N = 100
0.10
0.08
0.06
0.04
0.02
0.00
45
55
65
75
85
95
105
115
125

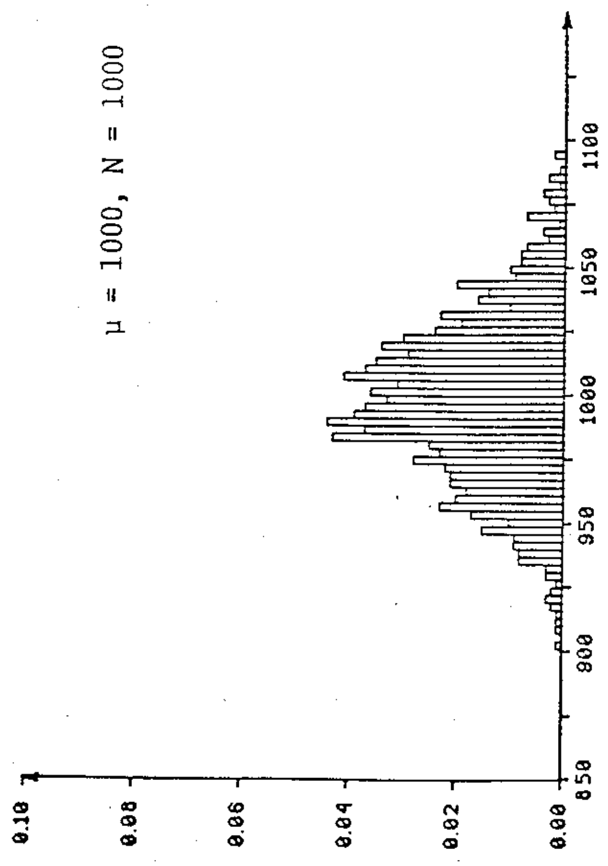

μ = 85, N = 1000
0.10
0.08
0.06
0.04
0.02
0.00
45
55
65
75
85
95
105
115
125

Menetelmä 2


μ = 1000, N = 300
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100


μ = 1000, N = 3000
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100


μ = 1000, N = 100
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100


μ = 1000, N = 1000
0.00
0.02
0.04
0.06
0.08
0.10
850
900
950
1000
1050
1100

```
3000 'POISGEN/BAS - POISSON-GENERAATTORIT BY RV 2.9.1983 *****
3010 'TESTATTAVAT GENERAATTORIT OVAT DIETERIN JA AHRENSIN
3020 'EKSAKTI MENETELMÄ (ACM TRANS. MATH. SOFTWARE VOL.8 NO.2,
3030 'JUNE 1982 S. 163-179), JA BEARDIN, PENTIKÄISEN JA PESOSEN
3040 'RISKITEORIAN 3. PAINOKSEN HARJ.TEHTÄVÄN 6.8.1
3050 'MALLIRATKAISUSSA ESIINTYVÄ APPROKSIMAATIOALGORITMI.
3060 DIM A(7),FA(9),PP(35),B(10),AA(100),KP(100):IN=1:P2=6.283185
3070 CLS:PRINT"POISSON-JAKAUMAGENERAATTORIT"
3080 PRINT"==========================="
3090 PRINT
3100 PRINT"OHJELMAN TARKOITUKSENA ON KOKEILLA JA VERTAILLA DIETERIN JA"
3110 PRINT"AHRENSIN TARKKAA MENETELMÄÄ (MENETELMÄ 1) JA BPP:N"
3120 PRINT"3. PAINOKSESSA ESITETTÄVÄÄ APPROKSIMAATIOMENETELMÄÄ"
3130 PRINT"(MENETELMÄ 2) POISSON-JAKAUTUNEIDEN SATUNNAISLUKUJEN GENEROIMISEKSI."
3140 PRINT"PIENILLÄ POISSON-PARAMETRIN ARVOILLA KÄYTETÄÄN SAMAA"
3150 PRINT"EKSAKTIA GENEROINTIALGORITMIA."
3160 PRINT
3170 INPUT"KUINKA MONTA POISSON-SATUNNAISLUKUA GENEROIDAAN";NO
3180 INPUT"TULOSTETAANKO SATUNNAISLUVUT (K/E)";K1$
3190 INPUT"KOOTAANKO NIISTÄ GRAAFINEN ESITYS (K/E)";K2$
3200 INPUT "ANNA POISSON-PARAMETRI";MU
3201 A(0)=-.5:A(1)=.3333333:A(2)=-.2500058:A(3)=.2000118
3202 A(4)=-.1661269:A(5)=.1421878:A(6)=-.1384794:A(7)=.1250060
3203 FA(0)=1:FA(1)=1:FA(2)=2:FA(3)=6:FA(4)=24:FA(5)=120
3204 FA(6)=720:FA(7)=5040:FA(8)=40320:FA(9)=362880
3205 Q1=.3989423:Q2=.04166667:Q3=.3:Q4=.1428571:Q5=.1069
3206 Q6=1.8:Q7=-.6744:Q8=.08333333:Q9=4.8:QA=.25:QB=.5:QC=.458:QD=50.5
3207 J=0:N=NO:SM=SQR(MU):W=INT(SM/10+.6):K5=INT(MU)-50*W:K6=INT(MU)+50*W
3210 INPUT"VALITSE GENERAATTORI 1/2";VA
3220 IF NOT(VA=1 OR VA=2)GOTO 3210
3230 IF VA=2 GOTO 3680
3240 PRINT"DIETERIN JA AHRENSIN NORMAALIEKSPONENTTIALGORITMI."
3245 LPRINT " "
3250 LPRINT"DIETERIN JA AHRENSIN NORMAALIEKSPONENTTIALGORITMI."
3290 PRINT"NO=";NO;"MU=";MU;TIME$:LPRINT"NO=";NO;"MU=";MU;TIME$
3300 FOR I%=1 TO NO:JJ=0:IF VA=3 GOTO 3510
3302 IF MU=MP GOTO 3320
3304 IF MU<10 GOTO 3510
3310 MP=MU:S=SQR(MU):D=6*MU*MU:L=INT(MU-1.1484)
3320 GOSUB 3810:G=MU+S*Z:IF G<0 GOTO 3350
3330 K=INT(G):IF K>=L GOTO 3590
3335 DI=MU-K:U=RND(0)
3340 IF D*U>=DI*DI*DI GOTO 3590
3350 IF MU=MO GOTO 3380
3355 MO=MU:OM=Q1/S:B1=Q2/MU
3360 B2=Q3*B1*B1:C3=Q4*B1*B2:C2=B2-15*C3
3370 C1=B1-6*B2+45*C3:C0=1-B1+3*B2-15*C3:C=Q5/MU
3380 IF G<0 GOTO 3400
3385 KF=0:GOTO 3440
3390 IF FY-U*FY<=PY*EXP(PX-FX)GOTO 3590
3400 U0=RND(0):E=-LOG(U0):U=RND(0):U=U+U-1
3410 T=Q6+E*SGN(U):IF T<=Q7 GOTO 3400
3420 K=INT(MU+S*T):DI=MU-K:KF=1:GOTO 3440
3430 IF C*ABS(U)>PY*EXP(PX+E)-FY*EXP(FX+E) GOTO 3400
3435 GOTO 3590
3440 IF K>=10 GOTO 3450
```

```
3445 PX=-MU:PY=MU*K/FA(K):GOTO 3490
3450 DE=Q8/K:DE=DE-Q9*DE*DE*DE:V=D1/K
3460 IF ABS(V)<=QA GOTO 3470
3465 PX=K*LOG(1+V)-D1-DE:GOTO 3480
3470 PX=K*V*V*(((((((A7*V+A6)*V+A5)*V+A4)*V+A3)*V+A2)*V+A1)*V+A0)-DE
3480 PY=Q1/SQR(K)
3490 X=(Q8-D1)/S:XX=X*X:FX=-Q9*XX
3500 FY=QM*(((C3*XX+C2)*XX+C1)*XX+C0):ON KF+1 GOTO 3390,3430
3510 MP=0:IF MU=MO GOTO 3530
3515 MO=MU:IF 1>=INT(MU) THEN M=1 ELSE M=INT(MU)
3520 L=0:P=EXP(-MU):Q=P:P0=P
3530 U=RND(0):K=0:IF U <= P0 GOTO 3550
3540 IF L=0 GOTO 3560
3545 JA=1:IF U<=QC GOTO 3550
3547 IF L<=M THEN JA=L ELSE JA=M
3550 FOR K=JA TO L:IF U<=PP(K) GOTO 3590
3555 NEXT:IF L=35 GOTO 3530
3560 L=L+1:FOR K=L TO 35:P=P*MU/K:Q=Q+P:PP(K)=Q
3570 IF U<=Q GOTO 3580
3575 NEXT K:L=35:GOTO 3530
3580 L=K
3590 IF K1$="K" GOSUB 3860
3595 IF K2$="K" GOSUB 3900
3600 NEXT I%
3610 IF K1$="K" GOSUB 4000
3620 PRINT"NO=";NO;"MU=";MU;TIME$:LPRINT"NO=";NO;"MU=";MU;TIME$
3625 IF K2$="K" GOSUB 4050
3630 INPUT"HALUATKO JATKAA GENEROINTIA (K/E)";K3$
3640 IF K3$="K" GOTO 3170
3650 INPUT"HALUATKO MUODOSTAA JA TALL. LEVYLLE TEOR. OIKEAT P-FREKVENSSIT (K/E)";K4$
3660 IF K4$="K" GOTO 4120
3665 END
3680 PRINT"GAMMA-APPROKSIMAATIOALGORITMI BY TP ET AL. (TAI INVERSIO)"
3682 LPRINT " "
3685 LPRINT"GAMMA-APPROKSIMAATIOALGORITMI BY TP ET AL. (TAI INVERSIO)"
3690 INPUT"VALITSE INVERSIOMENETELMÄ (PARAM. PIENI, ANNA 3) TAI APPROKS.M. (ANNA 4)";VA
3700 IF NOT(VA=3 OR VA=4) GOTO 3690
3710 IF VA=3 GOTO 3290
3720 'GAMMA-APPROKSIMAATIOALGORITMI.
3730 PRINT "N0=";NO;"MU=";MU;TIME$:LPRINT"NO=";NO;"MU=";MU;TIME$
3740 FOR I%=1 TO NO:JJ=0
3750 GOSUB 3810:K=(Z+6*SM-1/(6*SM))^3/(108*SM)-MU-Q8
3760 K=INT(K):IF K<0 GOTO 3750
3770 IF K1$="K" GOSUB 3860
3780 IF K2$="K" GOSUB 3900
3790 NEXT I%:GOTO 3610
3800 'NORMAALIGENERAATTORI (BOX-MULLERIN MENETELMÄ).
3810 IN=-IN:IF IN>0 GOTO 3840
3820 X1=RND(0):X2=RND(0):R=SQR(-2*LOG(X1)):V=P2*X2
3830 ZZ=R*COS(V):Z=R*SIN(V):RETURN
3840 Z=ZZ:RETURN
3850 'NORMAALIT TULOSTUSTOIMET 10:N ERISSÄ.
3860 IF J=10 LPRINT B(1);B(2);B(3);B(4);B(5);B(6);B(7);B(8);B(9);B(10):B(1)=K:J=1:JJ=1
3870 IF JJ=0 THEN B(J+1)=K:J=J+1
3880 RETURN
3890 'FREKVENSSITAULUKON AA PÄIVITYS.
```

```
3900 IF MU)50 GOTO 3940
3910 IF K)99 GOTO 3930
3920 AA(K+1)=AA(K+1)+1:RETURN
3930 N=N-1:PRINT"K=";K;" )99":RETURN
3940 IF K(K5 OR K)=K6 GOTO 3970
3950 I1%=INT((K-INT(MU))/W):AA(I1%+51)=AA(I1%+51)+1
3960 RETURN
3970 N=N-1:PRINT"K=";K;"KUVION ULKOPUOLELLA."
3980 RETURN
3990 'VIIMEISEN RIVIN TULOSTUS
4000 IF J=10 GOTO 4020
4010 FOR J1=J+1 TO 10:B(J1)=0:NEXT J1
4020 LPRINT B(1);B(2);B(3);B(4);B(5);B(6);B(7);B(8);B(9);B(10)
4030 RETURN
4040 'FREKV.TAULUKON AA NORMEERAUS JA TALLETUS LEVYLLE.
4045 'MYöS PATSASPIIRRON KESKIPISTEET LASKETAAN JA TALLETETAAN.
4050 FOR J=1 TO 100:AA(J)=AA(J)/N:KP(J)=INT(MU)+(J-DD)*W-0.5:NEXT
4060 INPUT"TALLETETTAVAN VEKTORIN NIMI (ILMAN LIITETTA)";VE$
4070 V1$=VE$+"/DAT:1":OPEN"O",1,V1$
4080 FOR J=1 TO 100:PRINT #1,AA(J):NEXT:FOR J=1 TO 100:PRINT #1,KP(J):NEXT:CLOSE
4090 PRINT "LEVYTALLETUS O.K. W=";W
4100 FOR J=1 TO 100:AA(J)=0:NEXT:RETURN
4110 'TEOR. OIKEIDEN FREKVENSSIEN LASKENTA JA TALLETUS.
4120 IF MU)50 GOTO 4170
4130 AA(1)=EXP(-MU):FOR J=2 TO 100:AA(J)=AA(J-1)*MU/(J-1)
4140 IF J)MU AND J(100 AND AA(J)(.00005 GOTO 4160
4150 NEXT J:GOTO 4230
4160 FOR J1=J+1 TO 100:AA(J1)=0:NEXT J1:GOTO 4230
4170 IF MU)88 GOTO 4220
4180 AB=EXP(-MU)
4190 FOR J=1 TO INT(MU)+49:AB=AB*MU/J
4200 IF J)=INT(MU)-50 THEN AA(J-INT(MU)+51)=AB
4210 NEXT J:GOTO 4230
4220 PRINT"FREKV.LASKENTA EI ONNISTU - TULISI ALIVUOTO.":GOTO 4280
4230 INPUT"TALLETETTAVAN EKSAKTIN VEKTORIN NIMI (ILM. LIIT.)";VE$
4240 V1$=VE$+"/DAT:1":OPEN"O",2,V1$
4250 FOR J=1 TO 100:PRINT #2,AA(J):NEXT:CLOSE
4260 PRINT"LEVYTALLETUS O.K."
4270 FOR J=1 TO 100:AA(J)=0:NEXT
4280 INPUT"HALUATKO PALATA GENEROINTIIN";K1$
4290 IF K1$="K"GOTO 3170
4300 END
```

[9] Rubinstein, R. Simulation and the Monte Carlo Method. John Wiley & Sons, New York 1981.

[10] Solvency of Insurers and Equalization Reserves, Vol. I (General Aspects), Vol. II (Risk Theoretical Model), eds. T. Pentikäinen and J. Rantala. Insurance Publishing Company Ltd. Helsinki 1982.

[11] Voutilainen, R. Generating Poisson Variates on a Microcomputer. Paper presented at EURO VI, sixth European Congress on Operations Research, Vienna, Austria, July 19-22, 1983.