



11. Simulointi

Sisältö

- Johdanto
- Liikenneprosessin reaalisatioiden tuottaminen
- Satunnaismuuttujan arvonta annetusta jakaumasta
- Tietojen keruu
- Tilastollinen analyysi

Tiedote

- Luennon tavoite
 - Esitellään simulointi yhtenä liikenneteorian työkaluna
 - Käydään lyhyesti läpi simulointiin liittyvät eri osa-alueet
- Liikenneteorian syventävä moduuli sisältää myös erillisen kurssin aiheesta
 - S-38.3148 Tietoverkkojen simulointi
 - Pakollinen Teleliikenneteorian syventävässä moduulissa
 - Esitiedot: S-38.1145 ja C/C++ -kielen tuntemus
 - Luennoidaan **vain** joka toinen vuosi (syytä huomioida opintojen suunnittelussa!)
 - Luennoidaan seuraavan kerran syksyllä 2006

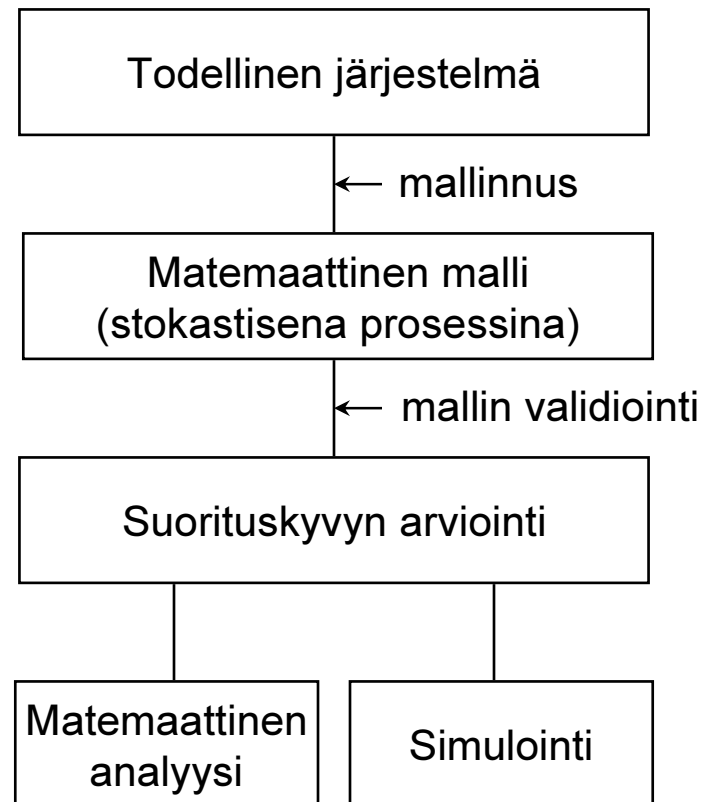
Mitä simulointi on?

- **Simulointi** on (liikenneteorian kannalta) eräs tilastollinen menetelmä tarkasteltavan järjestelmän suorituskyvyn arvioimiseksi
- Se sisältää neljä eri vaihetta:
 - Järjestelmän (olemassa olevan tai kuvitteellisen) mallinnus dynaamisena (ajassa kehittyvänä) stokastisena prosessina
 - Prosessin reaalisatioiden tuottaminen (“todellisuuden havainnointi”)
 - tällaista reaalisatiota kutsutaan usein simulointiajoksi (simulation run)
 - Tietojen keruu (“mittaus”)
 - Kerättyjen tietojen tilastollinen analyysi ja johtopäätösten teko

Vaihtoehto, mutta mille?

- Aiemmin olemme jo tutustuneet toiseen suorituskyvyn arviointimenetelmään, nimittäin **matemaattiseen analyysiin**
- Käsitelimme kaksi vaihetta
 - Järjestelmän mallinnus ajassa kehittyvänä stokastisena prosessina (tässä kurssissa rajoituimme syntymä-kuolema-prosesseihin)
 - Mallin analyyttinen ratkaisu
- Järjestelmän mallinnusvaihe on kummallekin yhteinen
- Tosin mallin tarkkuudella voi olla suuriakin eroja:
 - toisin kuin simulointi, matemaattinen analyysi edellyttää yleensä hyvinkin rajoittavien oletusten tekoa

Liikenneteoreettisen järjestelmän suorituskyvyn arviointi



Analyysi vs. simulointi (1)

- **Matemaattisen analyysin edut:**
 - Tulosten tuottaminen nopeaa (analyysin eli “yhtälöiden” jälkeen)
 - Tulokset tarkkoja
 - Antaa näkemystä
 - Optimointi usein mahdollista (vaikkakin saattaa olla vaikeaa)
- **Matemaattisen analyysin haitat:**
 - Asettaa rajoittavia ehtoja mallinnukseen
 - ⇒ malli yleensä liian yksinkertainen (esim. vain tasapainotila huomioitu)
 - ⇒ monimutkaisten järjestelmien suorituskyvyn arviointi lähes mahdotonta
 - Rajoittavien ehtojenkin vallitessa analyysi itsessään yleensä vaikeaa

Analyysi vs. simulointi (2)

- Simuloinnin **edut**:
 - Ei rajoittavia ehtoja mallinnusvaiheessa
 - ⇒ mahdollistaa monimutkaistenkin järjestelmien suorituskyvyn arvioinnin
 - Mallinnus yleensä hyvin suoraviivaista
- Simuloinnin **haitat**:
 - Tulosten tuottaminen yleensä työlästä (simulointiajot vaativat paljon prosessoriaikaa)
 - Tulokset epätarkkoja (tosin tarkentuvia: mitä enemmän ajoja, sitä tarkemmat tulokset)
 - Kokonaisnäkemys saaminen vaikeampaa
 - Optimointi mahdollista vain hyvin rajoitetusti (esim. muutaman erilaisen “parametrikombinaation” tai ohjausperiaatteen vertailu)

Stokastisen prosessin simuloinnin vaiheet

- Järjestelmän mallinnus ajassa kehittyvänä stokastisena prosessina
 - tästä on jo puhuttu kurssin aiemmilla luennoilla
 - jatkossa otamme lähtökohdaksi annetun mallin (so. stokastisen prosessin)
 - lisäksi rajoitamme tarkastelun tällä luennolla yksinkertaisiin liikenneteoreettisiin malleihin (vrt. aiemmat luennot)
- Prosessin reaalisatioiden tuottaminen
 - satunnaislukujen generointi
 - tapahtumaohjattu simulointi
 - usein simuloinnilla tarkoitetaan pelkästään tätä vaihetta (liikenneteorian kannalta se on kuitenkin simulointia suppeammassa mielessä)
- Tietojen keruu
 - transientti vaihe vs. tasapainotila
- Tilastollinen analyysi ja johtopäätökset
 - piste-estimaattorit
 - luottamusvälit

Simuloinnin toteutus

- Simulointi toteutetaan yleensä tietokoneohjelmana
- Simulointiohjelma sisältää yleensä kaikki edellä mainitut vaiheet mallinnusta ja johtopäätöksiä lukuunottamatta, ts.
 - järjestelmän malliksi valitun stokastisen prosessin reaalisatioiden tuottamisen,
 - tietojen keruun sekä
 - kerättyjen tietojen tilastollisen analyysin
- Simulointiohjelma voidaan toteuttaa
 - kokonaisuudessaan jollakin **yleiskäyttöisellä ohjelmointikielellä**
 - esim. C tai C++
 - joustavaa mutta työlästä ja riskialtista mahdollisille ohjelmointivirheille
 - käyttäen hyväksi joitakin **simulointiin erikoistuneita ohjelmakirjastoja**
 - esim. CNCL
 - erityisesti simuloiteja varten kehitetyillä **simulointiohjelmistoilla**
 - esim. OPNET, BONEs, NS (osittain perustuu o-kirjastoihin)
 - nopeaa ja luotettavaa (ohjelman laadusta riippuen) mutta jäykkää

Muita simulointitapoja

- Edellä kuvattu **diskreetti tapahtumapohjainen simulointi**
 - kyseessä **diskreetti, dynaaminen ja stokastinen** simulointi
 - eli miten simuloidaan tarkasteltavaa järjestelmää kuvaavan matemaattisen mallin (diskreettilaisen stokastisen prosessin) kehitystä ajassa tavoitena saada jotain tietoa ko. systeemin käyttäytymisestä
 - jatkossa rajoitumme tällaiseen simulointiin
- Muita simuloititapoja:
 - **jatkuvassa** simuloinnissa tila-avaruus on jatkuva (tilamuuttujien riippuvuudet annetaan yleensä differentiaaliyhtälösysteminä), esim. lentokoneen lentoradan simulointi
 - **staattisessa** simuloinnissa ajan kulumisella ei ole merkitystä (ei ole olemassa prosessia, jota luonnehtisi erilaiset tapahtumat), esim. moniulotteisten integraalien numeerinen integrointi ns. Monte-Carlo-menetelmällä
 - **deterministinen** simulointi ei taas sisällä ollenkaan satunnaisia komponentteja (esim. ensimmäinen esimerkki yllä)

Sisältö

- Johdanto
- Liikenneprosessin reaalisatioiden tuottaminen
- Satunnaismuuttujan arvonta annetusta jakaumasta
- Tietojen keruu
- Tilastollinen analyysi

Liikenneprosessin reaalisatioiden tuottaminen

- Oletetaan, että olemme mallintaneet tarkasteltavan järjestelmän stokastisena prosessina
- Seuraavana tehtävänä on prosessin reaalisatioiden tuottaminen
 - Se koostuu kahdesta osasta:
 - kaikille prosessin kulkuun vaikuttaville satunnaismuuttujille on arvottava arvot (yleensä reaaliluku) satunnaisesti ko. $sm:n$ jakaumasta ($sm:i$ en väliset riippuvuudet tietysti huomioiden)
 - näin saaduilla arvoilla konstruoidaan prosessin reaalisatio ts. sen kehittyminen ajassa
 - Nämä kaksi osaa eivät suinkaan tapahdu peräkkäin eri vaiheissa, vaan nimenomaan **limittäin** tai vuorotellen
 - Satunnaismuuttujien arvojen arvonta perustuu ns. **(pseudo)satunnaislukujen generointiin** (random number generation)
 - Prosessin reaalisatioiden konstruointi tehdään yleensä **tapahtumapohjaisesti** (discrete event simulation)

Tapahtumapohjainen simulointi (1)

- Idea: simulointi etenee **tapahdusta tapahtumaan**
 - jos jollakin aikavälillä ei tapahdu mitään, voimme hypätä ko. aikavälin yli
- Tapahtuma vastaa (yleensä) systeemin tilan muuttumista
 - esim. yksinkertaisessa liikenneteoreettisessa mallissa asiakkaiden saapumiset ja poistumiset systeemistä
 - tällaisia tapahtumia voidaan kutsua **perustapahtumiksi**
 - **ylimääräisiä tapahtumia** aiheutuu esim. tietojen keruusta ja prosessin reaalisaation generoinnin lopetuksesta
- Tapahtuma karakterisoidaan kahdella parametrilla
 - **tapahdusmomentti** (so. milloin tapahtuma käsitellään) ja
 - **tapahdustyyppi** (so. miten tapahtuma käsitellään)

Tapahtumapohjainen simulointi (2)

- Tapahtumat organisoidaan yleensä tapahtumahetken mukaan järjestetyksi **tapahtumalistaksi** (event list)
 - Kärjessä on seuraavaksi sattuva tapahtuma (siis aikaisin tapahtumahetki).
 - Listaa käydään läpi tapahtuma tapahtumalta (generoiden samalla uusia tapahtumia listan loppupäähän).
 - Kun tapahtuma on käsitelty, se poistetaan listalta.
- **Simulointikello** (simulation clock) kertoo, mikä on käsiteltävänä olevan tapahtuman hetki
 - se siis etenee hyppäyksittäin
- **Systemin tila** (system state) kertoo systeemin nykyisen tilan

Tapahtumapohjainen simulointi (3)

- Algoritmi yhden **simulointiajon** suorittamiseksi tapahtumapohjaisesti:
 - 1 Initialisointi
 - aseta simulointikello nolaksi
 - aseta systeemin tila valittuun alkuarvoonsa
 - generoi kunkin tapahtumatyyppin seuraava tapahtuma (mikäli mahdollista)
 - liitä näin saadut tapahtumat tapahtumalistaan
 - 2 Tapahtuman käsittely
 - aseta simulointiajaksi (tapahtumalistan kärjessä olevan) seuraavan tapahtuman tapahtumahetki
 - käsittele tapahtuma ja
 - generoi samalla uusia tapahtumia ja liitä ne tapahtumalistaan
 - päivitä systeemin tila
 - poista käsitelty tapahtuma tapahtumalistalta
 - 3 Lopetusehdon testaus
 - jos voimassa, lopeta tapahtumien generointi; muutoin palaa kohtaan 2

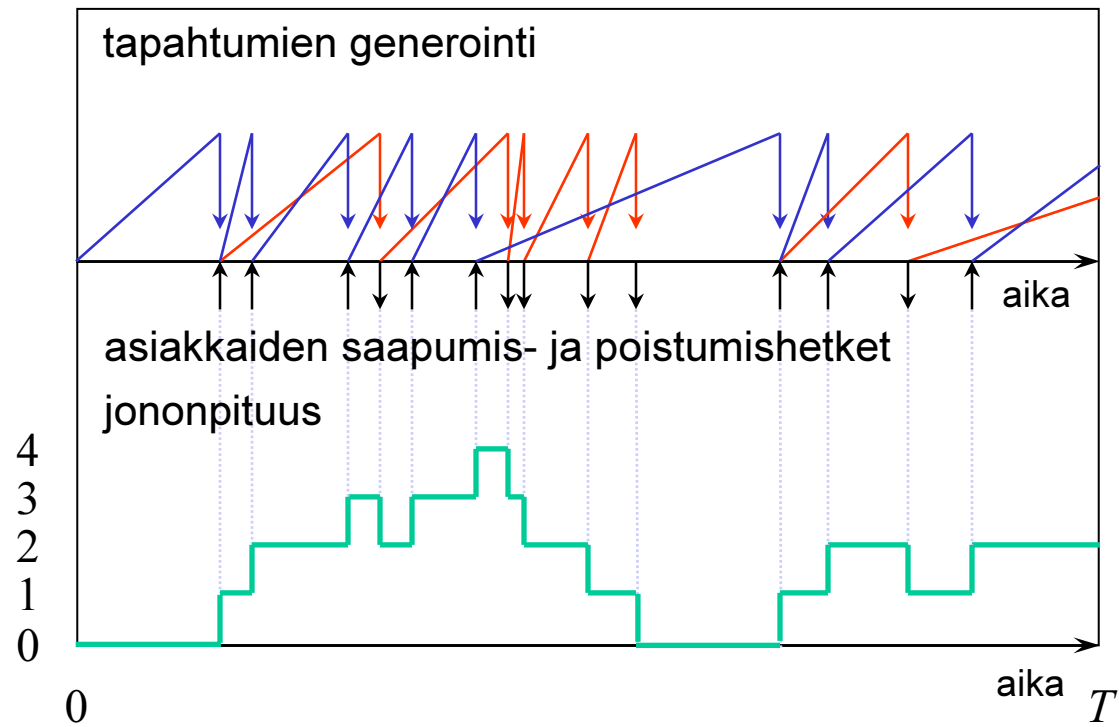
Esimerkki (1)

- **Tehtävä:** Simuloidaan M/M/1-jonon jononpituuden kehitystä ajassa hetkestä 0 hetkeen T olettaen, että systeemi on tyhjä hetkellä 0
 - Systeemin tila (hetkellä t) = jononpituus X_t
 - alkuarvo: $X_0 = 0$
 - Perustapahtumat:
 - asiakkaan saapuminen systeemiin
 - asiakkaan poistuminen systeemistä
 - Muut tapahtumat:
 - simuloinnin lopetus hetkellä T
- **Huom.** Tietojen keruuta ei ole sisällytetty tähän esimerkkiin

Esimerkki (2)

- Initialisointi:
 - asetetaan $X_0 = 0$
 - arvotaan ensimmäisen asiakkaan saapumishetki $\text{Exp}(\lambda)$ -jakaumasta
- Tapahtuman käsittely uuden asiakkaan saapuessa (hetkellä t)
 - systeemin tilaa eli jononpituutta kasvatetaan yhdellä: $X_t = X_t + 1$
 - jos systeemi oli tyhjä asiakkaan saapuessa, generoidaan ko. asiakkaan poistumishetki $t + S$, missä S on arvottu $\text{Exp}(\mu)$ -jakaumasta
 - generoidaan seuraavan asiakkaan saapumishetki $t + I$, missä I on arvottu $\text{Exp}(\lambda)$ -jakaumasta
- Tapahtuman käsittely asiakkaan poistuessa (hetkellä t)
 - systeemin tilaa eli jononpituutta vähennetään yhdellä: $X_t = X_t - 1$
 - jos systeemiin jäi asiakkaita, generoidaan seuraavaksi palveltavan asiakkaan poistumishetki $t + S$, missä S on arvottu $\text{Exp}(\mu)$ -jakaumasta
- Lopetusehto: $t > T$

Esimerkki (3)



Sisältö

- Johdanto
- Liikenneprosessin reaalisatioiden tuottaminen
- Satunnaismuuttujan arvonta annetusta jakaumasta
- Tietojen keruu
- Tilastollinen analyysi

Satunnaismuuttujan arvonta annetusta jakaumasta

- Pohjana ns. **(pseudo)satunnaislukujen generointi**
- Ensimmäinen askel
 - Tuottaa **riippumattomia** välillä 0 ja 1 **tasajakautuneita** satunnaismuuttujia (siis $U(0,1)$ -jakaumasta) käyttäen satunnaislukugeneraattoria
- Haluttuun jakaumaan päästään $U(0,1)$ -jakaumasta esimerkiksi jollakin seuraavista menetelmistä:
 - **uudelleenskaalaus** ($\Rightarrow U(a,b)$)
 - **diskretointi** ($\Rightarrow \text{Bernoulli}(p), \text{Bin}(n,p), \text{Poisson}(a), \text{Geom}(p)$)
 - **kertymäfunktion käännös** ($\Rightarrow \text{Exp}(\lambda)$)
 - **muut muunnokset** ($\Rightarrow N(0,1) \Rightarrow N(\mu,\sigma^2)$)
 - **hyväksymis-hylkäys-menetelmä** (kun kyseessä rajoitetulla välillä määritelty jatkuva jakaumaa, jolla rajoitettu tiheysfunktio)
 - tarvitaan kaksi riippumatonta $U(0,1)$ -jakaumaa noudattavaa sm:aa

Satunnaislukujen generointi

- **Satunnaislukugeneraattorilla** (random number generator) tarkoitetaan algoritmia, joka tuottaa sarjan (näennäisesti) satunnaisia kokonaislukuja Z_i jollakin välillä $0, 1, \dots, m-1$
 - tuotettu sarja on aina jaksollinen (tavoitteena mahdollisimman pitkä jakso)
 - generoidut luvut eivät tiukasti ottaen ole ollenkaan satunnaisia vaan täysin ennalta arvattavissa (tästä nimitys pseudosatunnainen)
 - jos satunnaislukugeneraattori on huolellisesti suunniteltu ja toteutettu, niin sen tuottamat pseudosatunnaiset luvut **kuitenkin näyttävät ikään kuin riippumattomilta ja samoin jakautuneilta (IID) noudattaen tasaista jakaumaa** joukossa $\{0, 1, \dots, m-1\}$
- Satunnaislukugeneraattorin generoimien satunnaislukujen “satunnaisuus” on todennettava tilastollisin testein
 - saadun empiirisen jakauman tasaisuus joukossa $\{0, 1, \dots, m-1\}$
 - generoitujen satunnaislukujen välinen riippumattomuus (käytännössä korreloimattomuus)

Satunnaislukugeneraattoreita

- **Lineaariset kongruentiaaliset generaattorit** (linear congruential generator).
 - yksinkertaisin
 - uusi satunnaisluku määräytyy algoritmisesti edellisestä, $Z_{i+1} = f(Z_i)$
⇒ jakso voi olla korkeintaan m
- Näistä erikoistapauksena saadaan ns. **multiplikatiiviset kongruentiaaliset generaattorit** (multiplicative congruential generator).
- Muita menetelmiä:
 - additive congruential generators, shuffling, ...

Linear congruential generator (LCG)

- **Lineaarinen kongruentiaalinen** satunnaislukugeneraattori tuottaa satunnaisia kokonaislukuja Z_i joukosta $\{0, 1, \dots, m-1\}$ kaavalla:

$$Z_{i+1} = (aZ_i + c) \bmod m$$

- parametrit a , c ja m ovat ei-negatiivisia kokonaislukuja ($a < m$, $c < m$)
- lisäksi tarvitaan ns. **siemenluku** (seed) $Z_0 < m$
- **Huom.**
 - Parametrit on valittava huolella; muutoin tuloksena kaikkea muuta kuin satunnaisia lukuja.
 - Tietyin edellytyksin jaksoksi saadaan maksimiarvo m
 - esim. kun m muotoa 2^b , c pariton ja a muotoa $4k + 1$ (b usein 48)

Multiplicative congruential generator (MCG)

- **Multiplikatiivinen kongruentiaalinen** satunnaislukugeneraattori tuottaa satunnaisia kokonaislukuja Z_i joukosta $\{0, 1, \dots, m-1\}$ kaavalla:

$$Z_{i+1} = (aZ_i) \bmod m$$

- parametrit a ja m ovat ei-negatiivisia kokonaislukuja ($a < m$)
- lisäksi tarvitaan siemenluku $Z_0 < m$
- **Huom.**
 - Kyseessä on siis LCG:n erikoistapaus valinnalla $c = 0$.
 - Parametrit on tässäkin tapauksessa valittava huolella
 - Mikään parametrikombinaatio ei tuota (maksimaalista) jaksoa m
 - esim. jos m muotoa 2^b , niin jakso on korkeintaan 2^{b-2}
 - Kuitenkin, jos m on **alkuluku**, jakso $m-1$ on mahdollinen
 - PMMLCG = prime modulus multiplicative LCG
 - esim. $m = 2^{31}-1$ ja $a = 16,807$ (tai $a = 630,360,016$)

U(0,1)-jakautuneen sm:n generointi

- Olkoon Z jonkin satunnaislukugeneraattorin tuottama (pseudo)satunnainen kokonaisluku välillä $\{0,1,\dots,m-1\}$
- Tällöin (approksimatiivisesti)

$$U = \frac{Z}{m} \approx U(0,1)$$

Tasajakaumaa noudattavan $sm:n$ generointi

- Olkoon $U \sim U(0,1)$
- Tällöin

$$X = a + (b - a)U \sim U(a, b)$$

- Tätä sanotaan **uudelleenskaalausmenetelmäksi** (rescaling method)

Diskreetin sm:n generointi

- Olkoon $U \sim U(0,1)$
- Oletetaan lisäksi, että Y on **diskreetti** sm
 - arvojoukolla $S = \{0,1,\dots,n\}$ tai $S = \{0,1,2,\dots\}$
- Merkitään $F(x) = P\{Y \leq x\}$. Tällöin

$$X = \min \{x \in S \mid F(x) \geq U\} \sim Y$$

- Tätä sanotaan **diskretointimenetelmäksi** (discretization method)
 - Itse asiassa kyseessä on ns. ‘kertymäfunktion käänös’-menetelmän eräs muoto
- **Esim.** Bernoulli(p)-jakauma:

$$X = \begin{cases} 0, & \text{jos } U \leq 1-p \\ 1, & \text{jos } U > 1-p \end{cases} \sim \text{Bernoulli}(p)$$

'Kertymäfunktion käännös' -menetelmä

- Olkoon $U \sim U(0,1)$
- Oletetaan, että Y on sellainen **jatkuva** sm, jolle kertymäfunktio $F(x) = P\{Y \leq x\}$ on aidosti kasvava
- Merkitään $F^{-1}(y)$:llä kertymäfunktion $F(x)$ käänteisfunktioita. Tällöin

$$X = F^{-1}(U) \sim Y$$

- Tätä sanotaan **'kertymäfunktion käännös'-menetelmäksi** (inverse transform method)
- **Tod.** Koska $P\{U \leq u\} = u$ kaikilla $u \in (0,1)$, pätee

$$P\{X \leq x\} = P\{F^{-1}(U) \leq x\} = P\{U \leq F(x)\} = F(x)$$

Eksponenttijakaumaa noudattavan sm:n generointi

- Olkoon $U \sim U(0,1)$
 - seuraus: $1-U \sim U(0,1)$
- Olkoon $Y \sim \text{Exp}(\lambda)$
 - kf $F(x) = P\{Y \leq x\} = 1 - e^{-\lambda x}$ on selvästikin aidosti kasvava
 - kf:n käänteisfunktio on $F^{-1}(y) = -(1/\lambda) \log(1-y)$
- Näin ollen ('kertymäfunktion käännös'-menetelmän mukaan)

$$X = F^{-1}(1-U) = -\frac{1}{\lambda} \log(U) \sim \text{Exp}(\lambda)$$

N(0,1)-jakautuneen sm:n generointi

- Olkoot U_1 ja U_2 **riippumattomia** ja samoin jakautuneita noudattaen $U(0,1)$ -jakaumaa
- Tällöin, ns. Box-Müller-menetelmän mukaan, alla annetut sm:t X_1 ja X_2 ovat myöskin **riippumattomia** ja samoin jakautuneita noudattaen $N(0,1)$ -jakaumaa:

$$X_1 = \sqrt{-2 \log(U_1)} \sin(2\pi U_2) \sim N(0,1)$$

$$X_2 = \sqrt{-2 \log(U_1)} \cos(2\pi U_2) \sim N(0,1)$$

Normaalijakaumaa noudattavan σ :n generointi

- Olkoon $X \sim N(0,1)$
- Uudelleenskaalausmenetelmällä saamme

$$Y = \mu + \sigma X \sim N(\mu, \sigma^2)$$

Sisältö

- Johdanto
- Liikenneprosessin reaalisatioiden tuottaminen
- Satunnaismuuttujan arvonta annetusta jakaumasta
- Tietojen keruu
- Tilastollinen analyysi

Tilastotietojen keruu

- Johdannossa otettiin lähtökohdaksi, että simuloinnin tavoitteena on tarkasteltavan järjestelmän suorituskyvyn arviointi. Simuloimalla siis pyritään arvioimaan jonkin suorituskykyyn liittyvän parametrin arvo α .
 - Tämä parametri voi liittyä joko järjestelmän **transienttiin** käyttäytymiseen tai sitten ns. **tasapainotilaan** (steady state)
 - Esim. 1 ja 2 (transientti käyttäytyminen)
 - k :n ensimmäisen asiakkaan keskimääräinen odotusaika M/M/1-jonossa olettaen, että systeemi on aluksi tyhjä
 - keskimääräinen jononpituus M/M/1-jonossa aikavälillä $[0, T]$ olettaen, että systeemi on aluksi tyhjä
 - Esim. 3 (tasapainotilanne)
 - keskimääräinen odotusaika M/M/1-jonossa tasapainotilanteessa
- Yksittäinen simulointiajo tuottaa yhden havainnon X , joka jollakin lailla kuvaa arvioitavaa parametria
- Tilastollisten päätelmien tekemiseksi tarvitsemme kuitenkin useita havaintoja X_1, \dots, X_n (mielellään IID)

Transienttien piirteiden simulointi (1)

- Esimerkki 1:
 - Tarkastellaan k :n ensimmäisen asiakkaan keskimääräistä odotusaikaa M/M/1-jonossa olettaen, että systeemi on aluksi tyhjä
 - Simulointia jatketaan, kunnes viimeinenkin näistä k asiakkasta on saapunut ja päässyt palveluun
 - Yksittäisestä simulointiajosta saatava havainto X on tässä tapauksessa näiden k asiakkaan odotusaikojen W_i keskiarvo ko. simulointiajossa:

$$X = \frac{1}{k} \sum_{i=1}^k W_i$$

- Riippumattomia ja samoin jakautuneita (IID) havaintoja X_1, \dots, X_n voidaan tuottaa ns. **‘riippumattomien toistojen’ -menetelmällä** (independent replications)
 - ts. tekemällä useita samanlaisia mutta toisistaan riippumattomia simulointiajoja (toisistaan riippumattomilla satunnaisluvuilla)

Transienttien piirteiden simulointi (2)

- Esimerkki 2:
 - Tarkastellaan keskimääräistä jononpituutta M/M/1-jonossa aikavälillä $[0, T]$ olettaen, että systeemi on aluksi tyhjä
 - Simulointia jatketaan ennalta määrättyyn hetkeen T asti
 - Yksittäisestä simulointiajosta saatava havainto X on tässä tapauksessa jononpituuden $Q(t)$ aikakeskiarvo yli välin $[0, T]$ ko. simulointiajossa:

$$X = \frac{1}{T} \int_0^T Q(t) dt$$

- Huom. Ko. integraali on helposti laskettavissa, koska jononpituus ei muutu tapahtumien välillä
- Riippumattomia ja samoin jakautuneita (IID) havaintoja X_1, \dots, X_n voidaan jälleen tuottaa 'riippumattomien toistojen'-menetelmällä

Tasapainotilaan liittyvien piirteiden simulointi (1)

- Tilastotietojen keruu yksittäisestä simuloinnista tapahtuu periaatteessa samalla tavalla kuin transientteja piirteitä simuloitaessa.
- Simuloinnin alussa on kuitenkin tyypillisesti ns. **lämmittelyvaihe** (warm-up phase), ennen kuin systeemi on likimain tasapainossa, mikä aiheuttaa
 - overheadia = “turhaa simulointia”
 - harhaisuutta estimaattiin
 - tarpeen määrittellä, kuinka pitkä lämmittelyvaihe tarvitaan
- Riippumattomien ja samoin jakautuneiden (IID) havaintojen X_1, \dots, X_n tuottamiseksi (ainakin likimain) on kaksi eri tapaa:
 - riippumattomat toistot (independent replications) ja
 - ns. ‘batch means’ -menetelmä

Tasapainotilaan liittyvien piirteiden simulointi (2)

- **Riippumattomien toistojen menetelmä:**
 - tehdään useita samanlaisia mutta toisistaan riippumattomia simulointiajoja (so. saman systeemin simulointia samasta lähtötilasta mutta toisistaan riippumattomilla satunnaisluvuilla)
 - kussakin ajossa tilastotietojen keruu aloitetaan vasta lämmittelyvaiheen jälkeen (kuten sanottu, oma ongelmansa on tämän lämmittelyvaiheen pituuden määrittäminen)
 - havainnot IID
- **'Batch means' -menetelmä:**
 - yksi (erittäin) pitkä simulointiajo, joka lämmittelyvaiheen jälkeiseltä osalta (keinotekoisesti) jaetaan n :ään yhtä pitkään jaksoon, joita tietojen keruun kannalta käsitellään omina simulointiajoinaan
 - tarvitaan vain yksi lämmittelyvaihe
 - mutta havainnot eivät ole enää täysin riippumattomia (eivätkä tarkkaan ottaen täysin samoin jakautuneitakaan)
 - mitä pitempi jakso (eli pienempi n), sitä riippumattomammat havainnot₃₈

Sisältö

- Johdanto
- Liikenneprosessin reaalisatioiden tuottaminen
- Satunnaismuuttujan arvonta annetusta jakaumasta
- Tietojen keruu
- Tilastollinen analyysi

Parametrien estimointi

- Kuten edellisessä kohdassa todettiin, simuloinnilla pyritään arvioimaan jonkin suorituskykyyn liittyvän parametrin arvo α
- Yksittäinen simulointiajo tuottaa kyseisestä parametrasta havainnon X_i , joka siis on satunnaismuuttuja
 - Havaintoa X_i sanotaan **harhattomaksi** (unbiased), jos $E[X_i] = \alpha$
- Olet. että havainnot X_i ovat IID keskiarvolla α ja varianssilla σ^2
 - Tällöin **otoskeskiarvo** (sample mean)

$$\bar{X}_n := \frac{1}{n} \sum_{i=1}^n X_i$$

- on parametrin α **harhaton** ja **tarkentuva** estimaattori, sillä

$$E[\bar{X}_n] = \frac{1}{n} \sum_{i=1}^n E[X_i] = \alpha$$

$$D^2[\bar{X}_n] = \frac{1}{n^2} \sum_{i=1}^n D^2[X_i] = \frac{1}{n} \sigma^2 \rightarrow 0 \quad (\text{kun } n \rightarrow \infty)$$

Esimerkki

- Pyrimme arvioimaan simuloimalla 25:n ensimmäisen asiakkaan keskimääräistä odotusaikaa M/M/1-jonossa kuormalla $\rho = 0.9$, kun systeemi hetkellä 0 on tyhjä.
 - Teoreettinen arvo: $\alpha = 2.12$ (ei triviaali)
 - Havainnot X_i kymmenestä simulointiajosta ($n = 10$):
 - 1.05, 6.44, 2.65, 0.80, 1.51, 0.55, 2.28, 2.82, 0.41, 1.31
 - Näin ollen parametrin α piste-estimaatti on

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i = \frac{1}{10} (1.05 + 6.44 + \dots + 1.31) = 1.98$$

Estimaattorin luottamusväli (1)

- **Määr.** Väliä $(\bar{X}_n - y, \bar{X}_n + y)$ sanotaan parametrin α **luottamusväliksi** (confidence interval) **luottamustasolla** (confidence level) $1-\beta$, jos

$$P\{|\bar{X}_n - \alpha| \leq y\} = 1 - \beta$$

- Tulkinta: “parametri α kuuluu ko. välille tn:llä $1-\beta$ ”
- Oletetaan sitten, että havainnot $X_i, i = 1, \dots, n$, ovat IID tuntemattomalla keskiarvolla α mutta **tunnetulla** varianssilla σ^2
- Keskeisen raja-arvolauseen mukaan (kts. Luento 5, kalvo 48), ainakin suurilla n :n arvoilla pätee

$$Z := \frac{\bar{X}_n - \alpha}{\sigma / \sqrt{n}} \approx N(0,1)$$

Estimaattorin luottamusväli (2)

- Merk. z_p :llä $N(0,1)$ -jakauman p -fraktiilia
 - ts. $P\{Z \leq z_p\} = p$, missä $Z \sim N(0,1)$
 - esim. $\beta = 5\%$ eli $1-\beta = 95\% \Rightarrow z_{1-(\beta/2)} = z_{0.975} \approx 1.96 \approx 2.0$
- **Väite.** Parametrin α luottamusväli luottamustasolla $1-\beta$ on

$$\bar{X}_n \pm z_{1-\frac{\beta}{2}} \cdot \frac{\sigma}{\sqrt{n}}$$

- **Tod.** Määritelmän mukaan pitää osoittaa, että

$$P\left\{|\bar{X}_n - \alpha| \leq z_{1-\frac{\beta}{2}} \cdot \frac{\sigma}{\sqrt{n}}\right\} = 1 - \beta$$

11. Simulointi

$$P\{|\bar{X}_n - \alpha| \leq y\} = 1 - \beta$$

$$\Leftrightarrow P\left\{\frac{|\bar{X}_n - \alpha|}{\sigma/\sqrt{n}} \leq \frac{y}{\sigma/\sqrt{n}}\right\} = 1 - \beta$$

$$\Leftrightarrow P\left\{\frac{-y}{\sigma/\sqrt{n}} \leq \frac{\bar{X}_n - \alpha}{\sigma/\sqrt{n}} \leq \frac{y}{\sigma/\sqrt{n}}\right\} = 1 - \beta$$

$$\Leftrightarrow \Phi\left(\frac{y}{\sigma/\sqrt{n}}\right) - \Phi\left(\frac{-y}{\sigma/\sqrt{n}}\right) = 1 - \beta \quad [\Phi(x) := P\{Z \leq x\}]$$

$$\Leftrightarrow \Phi\left(\frac{y}{\sigma/\sqrt{n}}\right) - (1 - \Phi\left(\frac{y}{\sigma/\sqrt{n}}\right)) = 1 - \beta \quad [\Phi(-x) = 1 - \Phi(x)]$$

$$\Leftrightarrow \Phi\left(\frac{y}{\sigma/\sqrt{n}}\right) = 1 - \frac{\beta}{2}$$

$$\Leftrightarrow \frac{y}{\sigma/\sqrt{n}} = z_{1-\frac{\beta}{2}}$$

$$\Leftrightarrow y = z_{1-\frac{\beta}{2}} \cdot \frac{\sigma}{\sqrt{n}}$$

Estimaattorin luottamusväli (3)

- Yleensä odotusarvon α lisäksi myös varianssi σ^2 on tuntematon
- Tällöin se pitää estimoida **otosvarianssista** (sample variance)

$$S_n^2 := \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X}_n)^2 = \frac{1}{n-1} (\sum_{i=1}^n X_i^2 - n\bar{X}_n^2)$$

- Voidaan osoittaa, että IID havainnoille otosvarianssi on todellisen varianssin σ^2 harhaton ja tarkentuva estimaattori:

$$E[S_n^2] = \sigma^2$$

$$D^2[S_n^2] \rightarrow 0 \quad (\text{kun } n \rightarrow \infty)$$

Estimaattorin luottamusväli (4)

- Oletetaan nyt, että havainnot X_i , $i = 1, \dots, n$, ovat IID noudattaen $N(\alpha, \sigma^2)$ -jakaumaa tuntemattomalla keskiarvolla α ja **tuntemattolla** varianssilla σ^2 . Tällöin voidaan osoittaa, että

$$T := \frac{\bar{X}_n - \alpha}{S_n / \sqrt{n}} \sim \text{Student}(n-1)$$

- Merk. $t_{n-1,p}$:llä Student($n-1$)-jakauman p -fraktiilia
 - ts. $P\{T \leq t_{n-1,p}\} = p$, missä $T \sim \text{Student}(n-1)$
 - esim. 1: $n = 10$ ja $\beta = 5\% \Rightarrow t_{n-1,1-(\beta/2)} = t_{9,0.975} \approx 2.26 \approx 2.3$
 - esim. 2: $n = 100$ ja $\beta = 5\% \Rightarrow t_{n-1,1-(\beta/2)} = t_{99,0.975} \approx 1.98 \approx 2.0$
- Näin ollen otoskeskiarvon luottamusväli luottamustasolla $1-\beta$ on

$$\bar{X}_n \pm t_{n-1,1-\frac{\beta}{2}} \cdot \frac{S_n}{\sqrt{n}}$$

Esimerkki (jatkoa)

- Pyrimme arvioimaan simuloimalla 25:n ensimmäisen asiakkaan keskimääräistä odotusaikaa M/M/1-jonossa kuormalla $\rho = 0.9$, kun systeemi hetkellä 0 on tyhjä.
 - Teoreettinen arvo: $\alpha = 2.12$
 - Havainnot X_i kymmenestä simulointiajosta ($n = 10$):
 - 1.05, 6.44, 2.65, 0.80, 1.51, 0.55, 2.28, 2.82, 0.41, 1.31
 - Otoskeskiarvo on 1.98 ja otoshajonta (eli otosvarianssin neliöjuuri) on

$$S_n = \sqrt{\frac{1}{9} ((1.05 - 1.98)^2 + \dots + (1.31 - 1.98)^2)} = 1.78$$

- Näin ollen parametrin α luottamusväli 95%:n luottamustasolla on

$$\bar{X}_n \pm t_{n-1, 1-\frac{\beta}{2}} \cdot \frac{S_n}{\sqrt{n}} = 1.98 \pm 2.26 \cdot \frac{1.78}{\sqrt{10}} = 1.98 \pm 1.27 = (0.71, 3.25)$$

Havaintoja

- Simulointikokeen tulos tarkentuu (so. piste-estimaatin luottamusväli kapenee), kun
 - simulointitoistojen eli riippumattomien havaintojen lukumäärää n kasvatetaan, tai
 - yksittäisen havainnon varianssia σ^2 pienennetään
 - esim. ajamalla pitempiä yksittäisiä simulointiajoja
 - muilla ns. 'varianssin reduktio' -menetelmillä
- Jos annettuna on haluttu simulointitulosten suhteellinen tarkkuus (so. otoskeskiarvon hajonnan ja odotusarvon välinen suhde), voidaan dynaamisesti päättää, kuinka monta riippumatonta simulointitoistoa on tehtävä ko. tavoitteeseen pääsemiseksi

Kirjallisuutta

- I. Mitrani (1982)
 - “Simulation techniques for discrete event systems”
 - Cambridge University Press, Cambridge
- A.M. Law and W. D. Kelton (1982, 1991)
 - “Simulation modeling and analysis”
 - McGraw-Hill, New York

THE END

