# Elinkaarimallin työllisyysasteen laskeminen Reinforced Learning-menetelmällä

Antti J. Tanskanen

30.12.2019

## Abstrakti

Elinkaarimallilla arvioimme työllisyysvaikutuksia elinkaarimallilla, joissa yksilöt optimoivat omaa hyötyään elinkaaren yli. Perinteisesti tällaisia malleja on ratkaistu dynaamisen ohjelmoinnilla tai maximum likehood-menetelmillä, mutta keinotekoisia neuroverkkoja funktioapprosimaattoreina käyttävät ”tekoälyratkaisut” mahdollistavat laajempien elinkaarimallien ratkaisemisen. Näytämme, että suuren osan koko sosiaaliturvaa ja työntekovalintoja sisältävä elinkaarimalli on mahdollista ratkaista malliriippumattomalla reinforced learning-menetelmällä. Ajamme mallin standardoidussa ympäristössä, jolloin tuleva algoritmikehitys on suoraan käytettävissä mallin ratkaisuun. Tässä artikkelissa kuvataan mallin rakenne.

# Johdanto

Elinkaarimalleja on käytetty Suomessa mm. eläkeuudistuksen vaikutusten arviointiin (Määttänen, 2013) ja (Hakola & Määttänen, 2005). Tässä tutkimuksessa tarkastellaan artikkeliin (Määttänen, 2013) pohjautuvaa elinkaarimallia. Mallia on laajennettu huomioimaan sosiaaliturvaa aiempaa laajemmin.

Elinkaarimallissa tarkastellaan henkilön elinkaarta pitkän aikaa, tässä tutkimuksessa 20-70-vuotiaana. Tällöin joka hetki vakuutettu arvioi, millaista nettotuloa työskentely tuottaa verrattuna sosiaaliturvaetuuksiin, ja tekee prioriteettiensä ja tulojen perusteella tekee päätöksen työskentelystään tai etuudella olosta. Mukana tarkastelussa tulee olla myös vapaa-ajan arvostus ja pidemmän aikavälin diskontatut hyödyt.

Työhön osallistumista voi pitää optimointiongelmana: vakuutettu maksimoi hyödyn nykyarvoa säätelemällä työhön osallistumista (Hakola & Määttänen, 2007). Optimointiongelmaan voi soveltaa esimerkiksi dynaamista ohjelmointia optimaalisen ratkaisun etsimisessä (Hakola & Määttänen, 2007).

## Tämä tutkimus

Tässä tutkimuksessa työhön osallistumispäätöksiä ratkaistaan Reinforced Learning-menetelmällä, joka oppii yrityksen ja erehdyksen kautta. Algoritmi yrittää erilaisia toimia, tarkastelee millaisia palkkioita niistä seurasi, ja sillä perusteella pyrkii parantamaan toimiaan. Rajatuissa ongelmissa menetelmällä päästään usein erittäin hyviin, ihmistason ylittäviin suorituksiin (esim. Karpathy, 2016).

Malli on rakennettu Suomen sosiaaliturvan ja verotuksen toteuttavan kirjaston päälle käyttäen standardoitua tekoäly-ympäristöä. Tavoitteena on toisaalta tarkastella politiikkavaikutusten työllisyysvaikutuksia ja toisaalta pyrkiä laajentamaan mallikirjoa, jota elinkaarimalleilla voi ratkaista.

# Menetelmät

## Tarkasteltu elinkaarimalli

Tarkastelemme artikkeliin (Määttänen, 2013) pohjautuvaa elinkaarimallia. Mallia on laajennettu, ja se ratkaistaan eri menetelmällä. Perusrakenne on kuitenkin sama: mallissa kuvataan yksilöiden työhönosallistumispäätöksiä heidän preferenssiensä perusteella.

## Yleinen rakenne

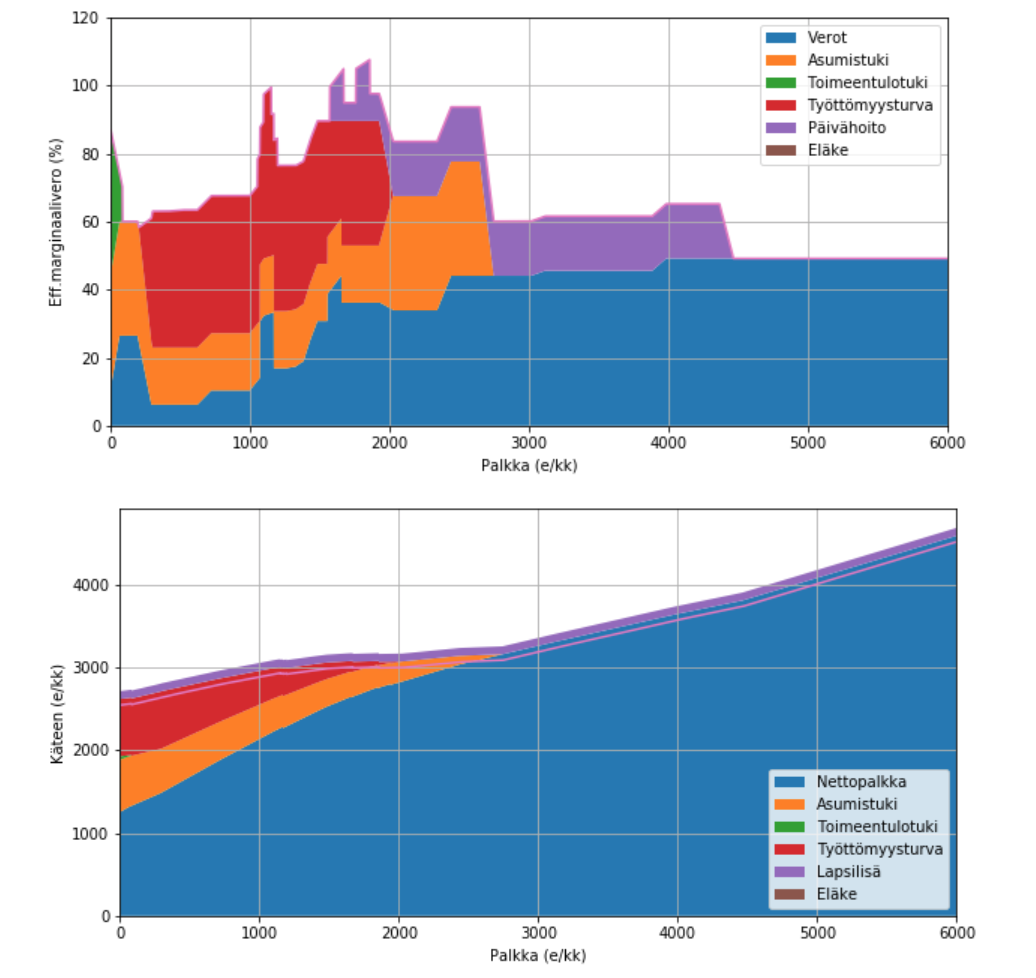
Elinkaarimalli on rakennettu modulaarisesti sillä ajatuksella, että mallia voi laajentaa ja muuttaa helposti. Tämä myös mahdollistaa hyvin helpon uusien mallinratkaisutapojen käyttöönoton.

Malli koostuu kolmesta modulista: (1) sosiaaliturva, (2) Gym-ympäristö ja (3) Elinkaarimalli. Modulit on toteutettu kolmella Python-kielisellä kirjastolla: etuudet ja verotuksen toteuttaa fin\_benefits-kirjasto (Tanskanen, 2019c), sosiaaliturvan OpenAI:n gym-ympäristöön toteuttaa econogym-kirjasto (Tanskanen, 2019b) ja varsinaisen elinkaarimallin toteuttaa lifecycle\_rl-kirjasto (Tanskanen, 2019a). Kaikki tutkimuksessa käytetyt koodit ja työkirjat asennusohjeineen ovat vapaasti saatavilla.

Kirjastojen vuorovaikutteinen käyttö onnistuu helposti Jupyter Notebook-työkirjojen avulla, mikä vaatii Pythonin asentamisen, tai Google Colab-palvelun (<https://colab.research.google.com/>) avulla, jolloin mitään malliin liittyvää ei tarvitse asentaa omalle tietokoneelle.

## Sosiaaliturva ja verotus

Sosiaaliturva on toteutettu tutkimusta varten kehitetyllä fin\_benefit -modulilla, jossa kuvataan tulot eri etuusmuodoilla, mm. ansiosidonnaisella työttömyysturvalla, toimeentulotuella ja asumistuella. Tässä tutkimuksessa sen avulla lasketaan käteen jäävää tuloa. Kirjaston avulla on myös mahdollista analysoida työnteon kannusteita (Kuva 1) hyvin vastaavasti kuin esimerkiksi tutkimuksessa (Viitamäki, 2015).



Kuva 1. fin\_benefits:in tuottamia kuvioita kannusteista esimerkkiperheelle, jossa on 2 aikuista ja 2 lasta. Lapsista 2 on päivähoidossa. Työtön (työmarkkinatuki), puoliso töissä (palkka 1250 e/kk). Asumismenot asumistuessa 980 e/kk

## Gym-ympäristö ja malliversiot

Jotta pystymme hyödyntymään OpenAI:n valmiita Reinforced Learning-algoritmeja, täytyy tarkastelukehikko upottaa OpenAI:n Gym-ympäristöön. Siinä on määritelty standardoitu liittymä Reinforced Learning-algoritmien ja ratkaistavan ympäristön välille, joka toteuttamalla on mahdollista hyödyntää valmiita algoritmeja. Tämän toteuttaa econogym-kirjasto.

OpenAI:n Gym-ympäristössä samasta ympäristöstä voi olla useita versiota. Nämä erotetaan toisistaan liittämällä -v0, -v1 tms mallin nimen loppuun. Tämän tutkimuksessa tarkastellaan useita versioita.

Perusversio on *unemployment-v0*, joka on yksinkertaistettu 3-tilainen ympäristö. Siinä agentti voi olla joko työssä, työtön tai vanhuuseläkkeellä. Mallissa on toteutettuna koko tällaisia henkilöitä koskeva sosiaaliturva fin\_benefits-modulin avulla.

Perusversiosta on EK:n työttömyysturvamallin mukainen versio on *unemploymentEK-v0*, jossa työttömyysturva on porrastettu, asumistukea ja toimeentulotukea on muokattu. Tämä versio on rakennettu *unemployment-v0*:n päälle viittaustekniikalla (luokkaa suoraan muokattu).

Laajempi perusversio on *unemployment-v1*, joka on yksinkertaistettu 13-tilainen ympäristö. Siinä agentti voi on olla myös kotihoidontuella, isyysvapaa, äitiysvapaalla (ml vanhempainvapaa), opiskelijana, työelämän ulkopuolella, työkyvyttömyyseläkkeellä, osatöissä, töissä vanhuuseläkkeellä tai osatöissä vanhuuseläkkeellä. Mukana on myös jako sukupuoliin ja kolmeen tulotasoon. Ympäristöstä on myös EK-versio *unemploymentEK-v1.*

Lisäksi mukana on porrastettu *unemploymentStep-v1*, malli, jossa *unemployment-v1*:tä on muokattu lisäämällä mukaan ansiosidonnaisen työttömyysturvan porrastus.

Esimerkkikoodina on myös *test\_environment-v0*, joka on valmiina templatena jatkokehitykselle. Mukana on myös alustava perustulo-ympäristö.

Mallista on helppo tehdä kaksi erilaista spesifikaatiota: stokastiset siirtymät ja deterministiset siirtymät. Näistä ajatuksellisesti lähempänä perinteistä ekonomistimallia ovat deterministiset siirtymät, joissa henkilö siirtyy siihen tilaan, jonka siirtymä todennäköisyys Markov-mallissa on suurin. Stokastisessa mallissa siirrytään Markov-prosessin mukaisesti. Molemmilla tavoilla on mahdollista mallintaa työllisyysastetta, erona on lähinnä se, että preferenssit kalibroidaan eri malleille eri tavalla, ja se että deterministiset siirtymät tuottavat saha-kuviota työllisyysasteisiin.

## Elinkaarimallin ratkaisu

Nimellä Reinforced Learning (RL) kutsutaan tutkimusaluetta, jossa pyritään tekoälyn avulla tekemään toimintapäätöksiä epävarmassa ympäristössä. Siinä toimija, esimerkiksi robotti, vuorovaikuttaa ympäristönsä kanssa, havainnoi toimintojensa seuraukset ja palkkiot, ja lopulta oppii havainnoistaan. Menetelmät ovat osin samoja ekonomistien agenttien käyttäytymisen tutkimukseen käyttämien kanssa, mm. dynaamista ohjelmointia pidetään RL-algoritmina.

Menetelmä perustuu Markov-päätösprosesseihin, joissa on kokoelma tiloja ja mahdollisia toimintoja niissä. Tila koostuu vektorista muuttujia, kuten ikä ja työllisyystilanne. Päätös toiminnosta, kuten työhön siirtyminen tai työttömyyteen siirtyminen, tilassa tehdään pelkästään tilan perusteella. Toiminta muuttaa tilavektoria Markov-prosessin mukaisesti stokastisesti.

Jokaiseen tilasiirtymään liittyy palkkio. Tavoitteena optimoinnissa on palkkioiden tarkasteluhetken pääoma-arvon maksimointi. Elinkaarimallissa (Määttänen et al, 2013) tarkasteltiin utiliteettejä, jotka voi suoraan tulkita RL-mallin palkkioksi: henkilön toiminnon seurauksena saama hyöty on nettotulojen logaritmi vähennettynä vapaa-ajanmenetyksen arvoa kuvaavalla vakiolla.

Jos tiloja on vähän, voi ongelman usein ratkaista melko helposti. Jos tiloja on paljon ja osa saa arvoja reaalilukujen joukossa, on tämä jo selvästi vaikeampaa. Tätä ”curse of dimensionality”:ä helpottamaan voidaan käyttää approksimatiivisia menetelmiä, kuten Deep Reinforced Learning-menetelmiä, joissa keinotekoisia neuroverkkoja käytetään ennustamaan tilojen arvofunktiota ja/tai tilasiirtymien arvofunktioita (Sutton & Barto, 2017).

RL on malliriippumaton algoritmi: samalla algoritmilla on mahdollista ratkaista niin tässä tarkasteltu elinkaarimalli kuin pelata yksinkertaisia konsolipelejä. RL-mallissa optimoidaan sekä politiikkaa että neuroverkolla tapahtuvaa utiliteettien nykyarvon approksimointi. Tämä kaksinkertainen optimointi voi aikaansaada ongelmia mallien konvergoinnissa. Tässä tutkimuksessa ongelmia tuli esille, kun yksilötason kuolleisuus huomioitiin mallissa.

Koneoppimisalgoritmit vaativat paljon opetusaineistoa. Vaikka ne ovat käytännössä tehokkaita, tunnetaan mallien konvergoitumisen teoreettisten ominaisuudet tunnetaan vielä huonosti (Agarwal et al., 2019).

Elinkaarimalli tarvitsee vähimmillään useita miljoonia havaintoja, jotta noin 99 prosenttia opetusaineiston varianssista selittyy. Esitetyt tulokset on laskettu mallilla, jota on sovitettu vähintään 10 miljoonaa havaintoa, jotta heilunta pienenee. Tulokset lasketaan 100 000 henkilön populaatiolle ja skaalataan koskemaan koko väestöä.

On tärkeää, että malli ehtii erittäin konvergoida riittävän lähelle globaalia optimia. Muutoin politiikkamuutosten arviointi mallilla on vaikeaa, koska mallin kuvaama käytös poikkeaa tavoitellusta, eivätkä tulokset ole uskottavia.

Mallin konvergenssia testattiin vertaamalla saatuja tuloksia dynaamiseen optimointiin, mutta tarkastelu tehtiin redusoidussa mallissa, jossa on vain viisi tilamuuttujaa. Tulokset vastasivat hyvin toisiaan.

Tarkastellusta mallista voi ajaa versiota, jossa ei ole stokastiikkaa alkavuuksissa lainkaan. Tällaisen mallin konvergenssi on selvästi parempaa kuin stokastisen mallin. Stokastisina ilmiöinä on toteutettu palkkaprosessi irtisanominen, kuolleisuus, ja lapsensaannin seurauksena tulevat vanhempainvapaat. Ongelmaksi näiden kanssa tulee ratkaisun konvergenssi, erityisesti jos kuolleisuus on mukana. Mallissa ei ole huomioitu yksilötason kuolleisuutta ennen 70 ikävuotta mukana, koska se aiheutti RL-menetelmän epästabiilisuutta. Tämä olisi helppo tehdä myös diskonttotekijän avulla, mutta ikävuosittaista diskonttokorkoa ei ole käytetyssä Stable baselines-kirjastossa toteutettuna.

Tarkasteltu elinkaarimalli on stokastinen siinä mielessä, että tilasiirtymät ovat stokastisia. Sovituksessa etsitään parhaat siirtymätodennäköisyydet tilojen välillä. Niiden avulla ennustamisessa sitten tehdään siirtymiä. Olisi myös mahdollista käyttää deterministisiä tilasiirtymiä, jolloin aina valittaisiin todennäköisin siirtymä (ok?), mutta stokastiikka tuottaa erityisesti pienillä tarkastelujoukoilla parempia tuloksia.

## Tulosprotokolla

Artikkelissa (Tanskanen, 2019d) esitetyt tulokset on ajettu protokollalla, jossa lähdetään satunnaisesta alkutilasta. Tätä mallia ajetaan 100 episodi kerrallaan, jonka jälkeen päivitetään neuroverkkojen painoja. Koko ajo kestää 20 miljoonaa askelta. Laajassa mallissa yksi episodi kestää 50 vuotta, missä on 200 askelta. Suppeassa mallissa yksi episodi kestää saman verran, mutta askelia on 50.

## Toteutus

Laskenta on toteutettu Python-kielisenä käyttäen hyväksi Open AI:n Gym-ympäristöä (Brockman et al., 2016). Open AI:n Gym on standardoitu ympäristö tekoälytutkimukseen, johon löytyy valmiita algoritmeja. Tässä tutkimuksessa on käytetty Stable Baselines-kirjaston (Hill et al., 2018) Deep Q-Learning-algoritmia (Wang et al, 2015) ja Actor-Critic Kronecker factored trust region-algoritmia (Wu et al., 2018). Jatkossa tarjolle tulee entistä parempia algoritmeja, jotka mahdollistava entistä laajemmat mallit. Tarkoituksena on myös tämän tutkimuksen jatkotyönä kehittää elinkaarimallien ratkaisuun soveltuvia algoritmejä.

Tässä tarkasteltu elinkaarimalli perustuu keinotekoisiin neuroverkkoihin, joten tavanomaiset neuroverkkojen sovitusta koskevat vaatimukset ja ongelmat pätevät. Tilamuuttujat on skaalattu niin, että ne pääsääntöisesti ovat välillä (-1,1), jolloin sovitus nopeutuu huomattavasti. Hyperparametrien optimoinnilla olisi jossain määrin parannettavissa sovitteita. Tässä on pääosin käytetty Stable baselines-kirjaston algoritmien vakioparametrejä, tosin sovitetussa neuroverkossa on 2x512x512 noodia). Liian suuri neuroverkko ylisovittaa, liian pieni taas ei pysty kuvaamaan ratkaisua. Ongelmana on myös se, että neuroverkkojen oppimisessa usein käytetyt menetelmät (esimerkiksi batch normalization ja drop out) eivät toimi kovin hyvin Reinforced Learning-malleissa.

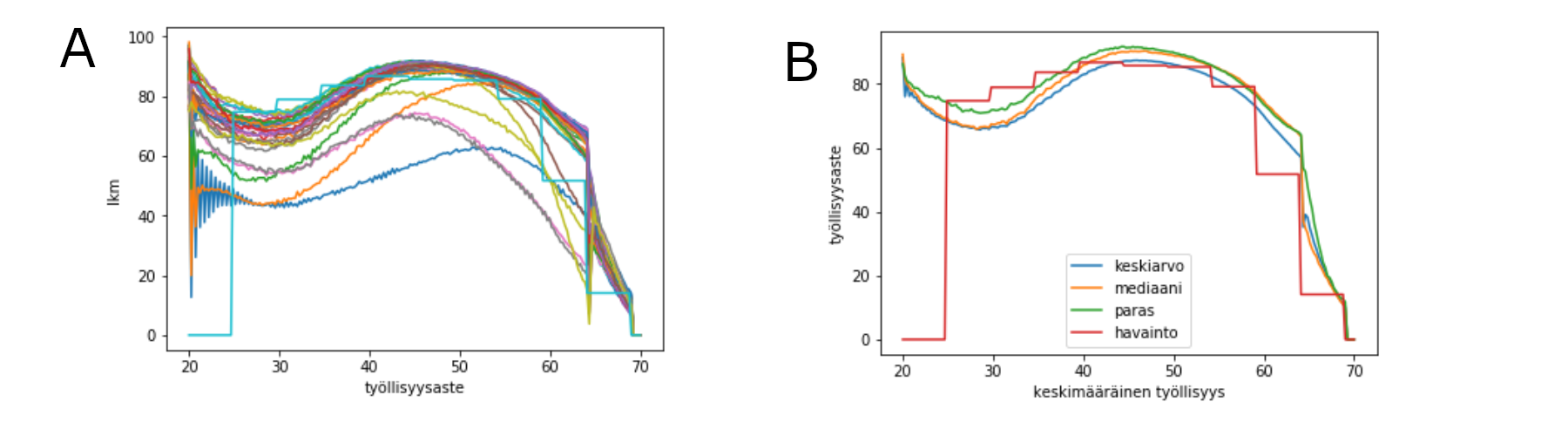
Markov-prosessien tilasiirtymät ovat stokastisia. RL-mallin sovitteet eivät aina konvergoi monotonisesti, vaan sovituksen kuluessa mallit saattavat myös siirtyä huonompiin politiikkoihin. Myöskään opitut toimet eivät ole täysin deterministisiä, joten eri ajokerroilla malli tuottaa hieman toisistaan poikkeavia tuloksia. Tästä syystä olemme (1) kiinnittäneet satunnaislukugeneraattorien siemenet, (2) ajaneet sovitteita useita kertoja ja käyttäneet keskiarvotuloksia, ja (3) käyttäneet vähintään 10 000 agenttia laskelmissa. Nämä toimet hidastavat tulosten saamista, mutta lisäävät niiden luotettavuutta.

Mallin ratkaiseminen ei välttämättä vaadi suurta konekapasiteettia. Tässä tutkimuksessa esitetyt laskelmat ovat täysin tehtävissä uudehkolla peruskannettavalla, ja sellaisella ne on tehtykin (Apple MacBook 15” vuosimallia 2015). Tulokset kuitenkin paranevat, mikäli enemmän laskentakapasiteettia on tarjolla. Laskenta myös skaalautuu erittäin hyvin lisäkapasiteetin mukana.

# Verifiointi

## Laaja malli

Tarkastellaan ensin sitä, kuinka pieniä muutoksia mallilla voi arvioida. Ajetaan sama malli 30 kertaa peräkkäin ja lasketaan tulosten erot. Kuva 1 näyttää, että huolimatta pitkästä sovituksesta Reinforced Learning voi tuottaa hyvin erilaisia tuloksia eri ajokerroilla. Tämä johtuu siitä, että optimointi on osin melko epästabiilia. Ongelman saa osin ratkaistua pitämällä kirjaa optimoinnin ajalta parhaasta sovitteesta ja käyttämällä sitä lopullisessa tuloksissa optimoinnin päätepisteen sijaan.



Kuva 1. Hajontaa työllisyysarviossa eri ajokerroilla. (A) 30:n ajon tuottamat ikäluokittaiset työllisyysarviot verrattuna havaittuun. (B) Ajoista lasketut työllisyysasteiden keskiarvo ja mediaani, ja parhaan kokonaishyödyn tuova ajo.

Kuvassa 1B verrataan mallin ennustamaan työllisyysastetta toteutuneeseen. Mallissa vanhuuseläkeikä on 65, kun taas aineistossa se on 63,5 (ok?). Käytetään jatkossa ylläkuvatulla tavalla saatua ennustetta.

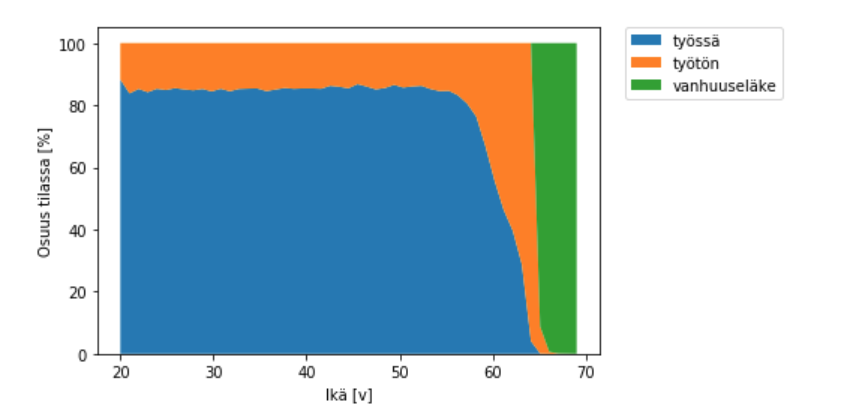
Työttömyys vastaa karkeasti tilastoja työttömyydestä. Työtähakevia työttömiä on noin 6 %, TEMin tilastoissa on noin 8 %, ja tämän lisäksi toimenpiteissä on noin neljännes työttömistä ja heidät tilastoidaan työvoiman ulkopuolelle. Kaikkiaan Suomessa laaja työttömyys on noin 11 prosenttia, mikä vastaa mallin antamaan tasoa.

Ikävälillä 25-40 työllisyysasteen notkahdus johtuu suurelta osin vanhempainvapaista, kotihoidontuesta, sekä pieneltä osin työkyvyttömyyseläkkeestä. 50-60 -vuotiaiden työllisyysasteen tippuminen johtuu suurelta osin työkyvyttömyyseläkkeestä, kun taas yli 60-vuotiaiden työllisyysasteen putoaminen johtuu pääosin työttömyysputkesta.

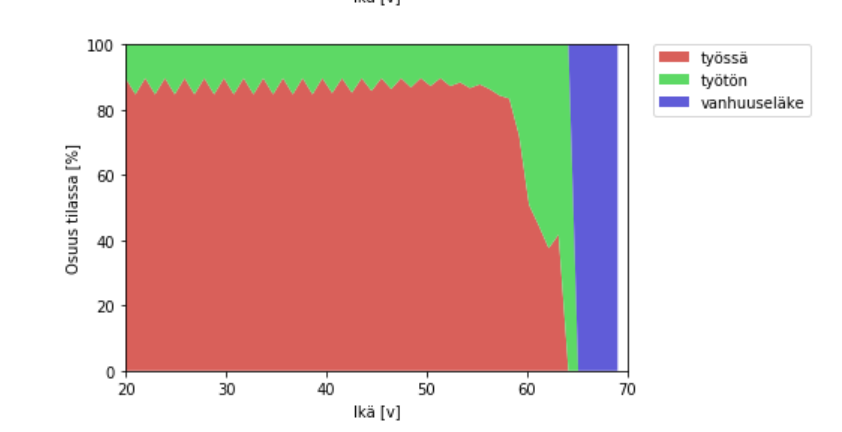
Lisää Laajan mallin tuloksia kts Tanskanen (2019d).

## Suppea malli

Vertailu dynaaminen optimointi ja RL *unemployment-v0* -mallin ratkaisussa.

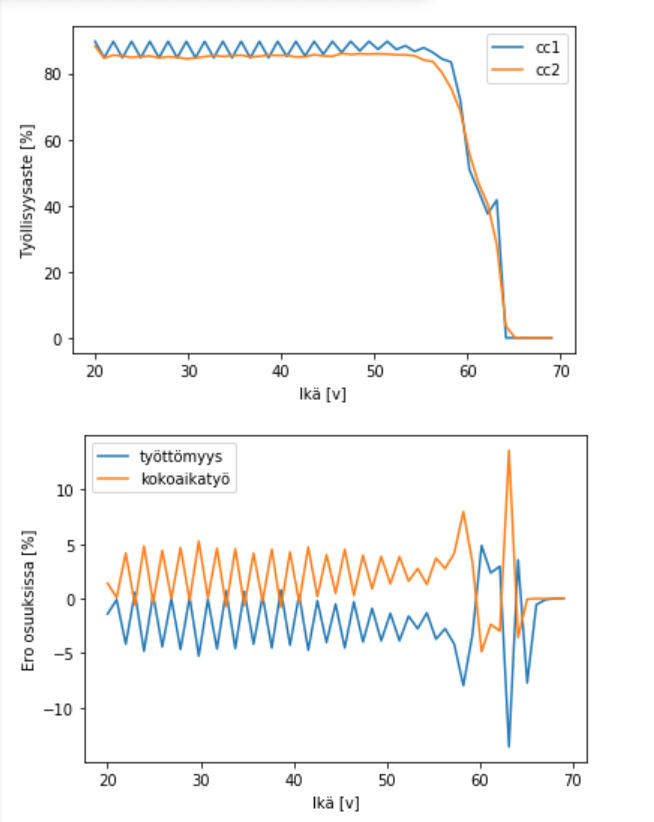


Kuva 5. Softmax



Kuva 6. Deterministic

Kuva 8. Dynaaminen optimointi



Kuva 7. Vertailu

# Diskussio

Elinkaarimalli on rakennettu modulaariseksi. Tämä mahdollistaa mallin helpon muokkaamisen, osin myös koska Python mahdollistaa objektien periytymisen ja jo aiemman koodin päälle rakentamisen. Esimerkiksi Ruotsin sosiaaliturva olisi melko helppo toteuttaa malliin korvaamalla *fin\_benefits* vastaavalla Ruotsin sosiaaliturvaa kuvaavalla mallilla.

# Kirjallisuus

Agarwal, Alekh, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. "Optimality and approximation with policy gradient methods in markov decision processes." arXiv preprint arXiv:1908.00261 (2019).

Arel, I., Deep Reinforcement Learning as Foundation for Artificial General Intelligence. In: Wang P., Goertzel B. (eds) Theoretical Foundations of Artificial General Intelligence. Atlantis Thinking Machines, vol 4. Atlantis Press, Paris. 2012.

Brockman, G., Cheung, V., Pettersson, L, Schneider, J,, Schulman, J., Zaremba, W. OpenAI Gym, arXiv:1606.01540, 2016.

Brown, N., Sandholm, T. Superhuman AI for multiplayer poker. Science 365, s. 885-890, 2019

Eerola, Essi. "Staattisissa vaikutusarvioissa pelataan nollasummapeliä." KAK 113: 263—265. 2017.

Elomäki, A, Ylöstalo, H (toim.), Tasa-arvoa talousarvioon – talousarvion sukupuolivaikutusten arviointi ja sukupuolitietoinen budjetointi. Valtioneuvoston selvitys- ja tutkimustoiminnan julkaisusarja 58/2018, 2018.

Finanssivalvonta & Kela: Suomen työttömyysturvaetuuksien kuukausitilasto helmikuussa 2019

Hakola, Tuulia, and Niku Määttänen. Vuoden 2005 eläkeuudistuksen vaikutus eläkkeelle siirtymiseen ja eläkkeisiin: arviointia stokastisella elinkaarimallilla. Eläketurvakeskus, 2007.

Hessel, M. Rainbow: Combining Improvements in Deep Reinforcement Learning. arXiv:1710.02298 (cs.AI), 2017

Hill, Ashley, and Raffin, Antonin and Ernestus, Maximilian and Gleave, Adam and Kanervisto, Anssi and Traore, Rene and Dhariwal, Prafulla and Hesse, Christopher and Klimov, Oleg and Nichol, Alex and Plappert, Matthias and Radford, Alec and Schulman, John and Sidor, Szymon and Wu, Yuhuai,

Stable Baselines, Github repository, <https://github.com/hill-a/stable-baselines>, 2018.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. Neural networks, 4(2), 251-257.

Igami, M. (2017). Artificial intelligence as structural estimation: Economic interpretations of Deep Blue, Bonanza, and AlphaGo. arXiv preprint arXiv:1710.10967.

Jiang, Z., Xu, D., & Liang, J. (2017). A deep reinforcement learning framework for the financial portfolio management problem. arXiv preprint arXiv:1706.10059.

Karpathy, A., Deep Reinforcement Learning: Pong from Pixels. 2016 http://karpathy.github.io/2016/05/31/rl/

Kober, Jens; Bagnell, J. Andrew; Peters, Jan. Reinforcement learning in robotics: A survey. The International Journal of Robotics Research, 2013, 32.11: 1238-1274.

Kotamäki, M, Työllistymisveroasteet Suomessa, Valtiovarainministeriön keskustelualoite 1/2014

Kotamäki, M, Mattila, J, Tervola, J. Distributional Impacts of Behavioral Effects – Ex-Ante Evaluation of the 2017 Unemployment Insurance Reform in Finland. Microsimulation, 2018

Kyyrä, Tomi; Matikka, Tuomas; Pesola, Hanna. Työttömyysturvan suojaosa ja työttömyyden aikainen työskentely. 2018.

Kyyrä, Tomi, 2017

Kärkkäinen, O, Tervola, J. Talouspolitiikan vaikutukset tuloeroihin ja työllisyyteen 2015–2018, Valtioneuvoston selvitys- ja tutkimustoiminnan julkaisusarja 59/2018, 2018.

Määttänen, Niku. "Eläkepoliittisten uudistusvaihtoehtojen arviointia stokastisen elinkaarimallin avulla." Kirjassa Lassila, Määttänen ja Valkonen (2013): Eläkeiän sitominen elinaikaan–miten käy työurien ja tulonjaon (2013).

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

Mårtensen, Pissaredes, 1998.

Ostbaum, M., Palkkojen muutosten vaikutus työllisyyteen, Kansantaloudellinen aikakauskirja 4/2017: ..

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... & Lillicrap, T. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science, 362(6419), 1140-1144.

Silver, D., Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, Demis Hassabis Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm, <https://arxiv.org/abs/1712.01815>, 2017

Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018 http://incompleteideas.net/book/bookdraft2017nov5.pdf

Tanskanen, Antti. Elinkaarimalli. Github repository. <https://github.com/ajtanskanen/lifecycle_rl> 2019a

Tanskanen, Antti. Suomen sosiaaliturva ja verotus Gym-ympäristönä. Github repository. <https://github.com/ajtanskanen/econogym> 2019b

Tanskanen, Antti. Suomen sosiaaliturva ja verotus Python-toteutuksena. Github repository. https://github.com/ajtanskanen/benefits 2019c

Tanskanen, Antti, et al. Unelmoivatko robotit ansiosidonnaisesta sosiaaliturvasta? Submitted to KAK, 2019d

Tikanmäki, Heikki, Hannu Sihvonen, and Janne Salonen. "Eläketurvakeskuksen ELSI-mikrosimulointimallin kuvaus." (2014).

Tilastokeskus: SISU-malli – Käyttöopas tulonsiirtojenja verotuksen mikrosimulointiin. Helsinki, 2016. <http://www.tilastokeskus.fi/static/media/uploads/tup/mikrosimulointi/sisu_kasikirja_2016.pdf>

Työllisyyspaketti-työryhmän esitykset. Työ- ja elinkeinoministeriön julkaisuja, Työelämä, 33/2016

Viitamäki, Heikki. "Työnteon kannustimet-mitä jää käteen?." (2015).

Virrankoski, J, Dale Mortensen, Christopher Pissarides ja markkinoiden etsintäkitkojen tutkimus, KAK 1/2011: 99—103.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Freitas, N. (2015). Dueling network architectures for deep reinforcement learning. arXiv preprint arXiv:1511.06581.

Wu, Y., Mansimov, E., Grosse, R. B., Liao, S., & Ba, J. (2017). Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In Advances in neural information processing systems (pp. 5279-5288).

Ziebart, B. D., Maas, A., Bagnell, J. A., & Dey, A. K. (2008). Maximum entropy inverse reinforcement learning.