Viikkoraportti 4

6.2.2015
Kirjoitin testit matriisikertolaskussa käytettäville apufunktioille ja strassen-algoritmille. Aloitin tutustumisen determinantin laskemisen mahdollisiin toteutuksiin. LU ja LUP-hajotelma vaikuttavat melko monimutkaisilta, enkä olekkaan varma, miten niitä pitäisi soveltaa determinantin selvittämisessä. Konsultoin matematiikkaa opiskelevaa kaveriani, mutta en päässyt asiasta vielä jyvälle. Toteutan ensin raa'an voiman ratkaisun determinantin laskemiseen ja ajan salliessa etsin nopeamman algoritmin toteutettavaksi. KIrjoitin aluksi testejä GNU Octavesta tarkistetuille tapauksille.

7.2.2015
Kirjoitin huolittelemattomalla clojurella (refaktoroin myöhemmin) O(n!) ajassa toimivan funktion determinantin selvittämiseksi. Funktio ratkoi melkein kaikki determinantit oikein, mutta suuremmilla matriiseilla (JVM:n) Integer pyörähti ympäri. Myöskin joillekkin pienemmille neliömatriiseille funktioni palautti eri arvon, kuin GNU Octave. Yritin selvittää vikaa, mutta en löytänyt vielä.

11.2.2015
Aloitin debuggaamalla virheellisesti toimivaa determinantti-funktiota, mutta siitä ei tullut mitään tänään. Tein viikon tehtäviin kuuluvan koodikatselmoinnin (https://github.com/nonpop/2015-periodi-3/issues/5). Tutustuin ohjelmakoodiin ja yritin ymmärtää sitä. Koodi oli hyvin luettavaa, mutta itse algoritmit niin laajoja ja minulle entuudestaan tuntemattomia. En osannut etsiä algoritmeista ongelmakohtia, sillä en saanut ajettua ohjelmaa enkä testejä. Koodikatselmuksen tehtävänannossa kehotettiin sivuttamaan koodin ulkoasu, mutta ainakin omassa tapauksessa se oli ainoa, mihin pystyin ottamaan kantaa.

12.2.2015
Pitkää kestäneen debuggaamisen jälkeen löysin determinantti-funktiosta kirjoitusvirheen (i ja 1 sekaisin). Opin samalla käyttämään clojuren debuggaus-työkaluja. Clojure kielenä tukee monennäköistä vianetsintää; editorissa pyörivän (esim Netbeans tai IntellijIDEA) Debug-tilan sijaan virheitä etsitään REPL-konsolista. REPL(Read-eval-print-loop) on komentorivityökalu, joka tulostaa sille annetun koodin pätkän tuloksen saman tien. Clojure-vianetsinnässä käteväksi osottautui makro, joka tulostaa tietyn funktion syötteen ja tulosteen joka kutsulla. Sen avulla vakuutuin, että apufunktioni toimivat kuten pitää kaikissa tapauksissa ja onnistuin rajaamaan ongelma-aluetta.
Kirjoitin determinantti-toiminnallisuuden apufunktioille pari testiä. Nostan koodikattavuuden taas ylös ensi viikolla ja dokumentoin uuden toiminnallisuuden. Tulen huomenna käymään pajassa konsultoimassa, kannattaisiko minun käyttää loppu kurssi vielä uuden algoritmin kirjoittamiseen vai käyttöliittymään ja dokumentteihin.

Koodin rivikattavuus raportti kansiossa /matrix-calculator/target/coverage.