Differentiaaliyhtälöitä, ohjelmointia ja delfiinimallinnusta

Heikki Apiola
Aalto yliopisto, matematiikan ja systeemianalyysin laitos, lehtori emeritus
heikki.apiola@aalto.fi

Aluksi

Käsillä on yritys hahmotella matemaattisen teorian, numeeristen menetelmien ja graafisen havainnollistuksen avulla tieteellisen laskennan aluetta, jossa mallintamisen työkaluina toimivat differentiaaliyhtälöt.

Differentiaaliyhtälöiden käsittelyyn on tietotekniikan ansiosta tullut viimeisten vuosikymmenien aikana uusia näkökulmia ja menetelmiä. Voidaan erottaa kolme erilaista, toisiaan täydentävää osaa:

- Analyyttinen
- Geometrinen, kvalitatiivinen
- Numeerinen

Näiden näkökulmien välinen vuorovaikutus on mitä hedelmällisin. Kaksi ensimmäistä antaa näkemystä ratkaisujen ja ratkaisuparvien ominaisuuksiin, pitkän aikavälin käyttäytymiseen ym. ja kolmas on se, jolla tulokset viime kädessä saadaan numeeriseen muotoon. Kuitenkin numeerinen ratkaisu yksinään on "sokea", se ei kerro koko ratkaisukentän käyttäytymisestä eikä tunnista joidenkin ratkaisujen luonteeseen vaikuttavien parametrien, kutsuttakoon vaikka "ominaisarvoiksi", vaikutusta. Kaikkia siis tarvitaan.

Kirjoituksen voidaan katsoa kuuluvan *tieteellisen laskennan* alueeseen, jossa rakennetaan siltaa matemaattisen teorian ja tietotekniikan välille. Käytössämme on ohjelmaympäristö, joka mahdollistaa numeerisen mallin suhteellisen vaivattoman muodostamisen graafisine havainnollistuksineen. Toisaalta sen avulla voi halutessaan opetella ohjelmoinnin perusteita korkean tason "funktionaalisella vektorikielellä". Tällainen on vapaasti ladattava Octave, johon olen viitannut aiemmissakin kirjoituksissani Solmussa. Octave:n kieli on käytännöllisesti katsoen identtinen kaupallisen Matlab-ohjelmiston kielen kanssa. Todettakoon kohtuuden nimissä, että kieli ja ympäristö on Matlab:n kehittäjän *Cleve Molerin* ja hänen "joukkueensa" *MathWorks Inc'n*: https://se.mathworks.com/ käsialaa ja innovaatiota. Korostan Octavea siksi, että sen ääreen lukija pääsee vapaasti.

Yhtenä tavoitteena on opettaa lukijalle differentiaaliyhtälöiden ratkaisemisen perustekniikoita, ihan käsin laskienkin. Niitä sovelletaan moninaisiin esimerkkeihin ja havainnollistetaan kuvin, joita lukija voi ohjeiden mukaan tuottaa ja muunnella. Henki on sellainen, että asiat esitetään varsin perusteellisesti ajatellen lukijaa, joka lähtee matkalle vaatimattomin esitiedoin.

Tarinan juoni

Seuraillaan delfiinien seikkailuja Välimeressä Aiolian saaristossa mallintaen populaation dynamiikkaa asteittain tarkentuvilla malleilla. Mallien mutkistuessa tarvitaan uusia ratkaisu- ja havainnollistusmenetelmiä, joita

esitellään vaiheittain, ja samalla kehitellään ratkaisuissa tarvittavia algoritmejä ohjelmakoodiin saakka.

Esiintyvät datat ja matemaattiset mallit ovat viitteestä [QG]. Ohjelmat ovat osittain saman viitteen pohjalta muunneltuja, osittain taas omia kehitelmiäni.

Delfiinijuonen ohella esitellään kolmea mainittua differentiaaliyhtälöiden näkökulmaa pohjautuen mm. mainioon viitteeseen [HW] sekä omiin vuosien varrella kehittämiini opetusmateriaaleihin. Käväisy symbolilaskennan puolella ja siinä yhteydessä pohdinta analyyttisten ratkaisujen "suljetun muodon" yleistyksestä on pienenä sivujuonteena mukana.

Yleisiä ratkaisujen **olemassaolo- ja yksikäsitteisyyslauseita** kosketellaan lyhyesti kevyellä otteella *vuotavaa ämpäriä* mallittavalla esimerkillä valaisten.

Myös pienimuotoista **ohjelmankehitystä** tarjoilen juonen edetessä aiheesta kiinnostuneille.

Mitä pohjatietoja tarvitaan?

- **Derivaatta** on perusta, jolle differentiaaliyhtälö rakentuu. Sitä ilman ei oikein voida elää. Tavalliset matemaattiset funktiot, kuten sini, kosini, eksponenttifunktio, logaritmi derivaattoineen on hyvä tuntea.
- **Integraalia ei** tarvitse tuntea, siitä selitetään kaikki, mitä tässä tarvitsee tietää, ja se on yhdellä sanalla sanottu: *antiderivaatta*.
- **Ohjelmointia ei** myöskään tarvitse osata.

Kirjoituksesta saat enemmän irti ottamalla OCTAVE-ohjelmiston käyttöön. Jos haluat rajoittua minimiin "kieliopinnoissa", riittää ladata viitteen https://matematiikkalehtisolmu.fi/2021/1/skriptit/ tiedostot ja suorittaa ne neuvottavalla tavalla.

Octaven asennus ja ohjeita

Sivulla www.octave.org annetaan latauslinkki ja peruskäyttöesimerkkejä.

Tässä pari laajempaa käyttöopasta:
https://octave.org/doc/v5.1.0/
http://www-h.eng.cam.ac.uk/help/programs/octave/tutorial/
Kts. myös suomenkielinen [AL] alla viitteissä.

Lisää Googlella esim. hakusanoilla *octave tutorial, matlab tutorial*.

Asennus on helpointa (Ubuntu) Linux:lla. Komentoikkunassa kirjoitat:
`sudo apt-get install octave`

Suositus: Octave-online

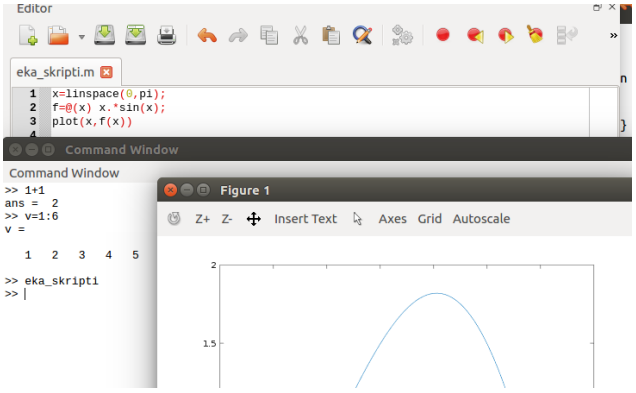
Voit välttyä asennusvaivoilta kirjautumalla Octave online -sivulle, josta pääset käyttämään ohjelmaa suoraan selaimella "käyttiksestä" riippumatta.

Kun klikkaat linkkiä https://octave-online.net/, voit heti ryhtyä kirjoittamaan komentoja alimpana näkyvälle komentoriville. Kannattaa ehdottomasti kirjautua, muussa tapauksessa käyttö on liian rajoitettua.

Ensiaskeleet

Jos olet asentanut OCTAVE:n koneellesi, voit nähdä seuraavanlaiset ikkunat, joiden oletusasettelu voi näyttäytyä hiukan eri muotoisena. *Komentoikkunaan* voit kirjoittaa suoraan komentoja kuvan mukaisesti. Vähänkin laajempaa hommaa tehdessäsi käytät editoria, johon talletetut komennot käynnistyvät kuvakkeesta tai kirjoittamalla komentoikkunaan `>> omaskripti`, jos komennot on talletettu tiedostoon `omaskripti.m`.

OCTAVE:n käyttöliittymä: Kaksi pääikkunaa: *komentoikkuna* ja *editori*.



Jos käytät *Octave-online* -ohjelmaa, näyttö jakaantuu kahteen osaan. Komentoikkunana toimii oikean puolen alin rivi, joka näkyy kuvassa. Aiemmat komennot tuloksineen ja mahdollisine kuvineen näkyvät komentorivin yläpuolella, ja aiempia komentoja voi selata ja editoida komentorivillä ↑-näppäimellä (vanhaan hyvään MATLAB-tyyliin).

Näin voit toimia kirjautumatta.

Kun kirjaudut, saat vasempaan ruutuun editorin. Tiedostot tallentuvat Octave-palvelimelle, voit myös tallentaa omalta koneeltasi palvelimelle ("upload"). Tässä

vasemman ikkunan yläosa, josta [+]-kuvakkeella voit avata editorin. Oikealla näkyy `RUN`, jolla skriptin voi ajaa.

Kannattaa huomata, että kaikki kooditiedostot ovat tavallisia tekstitiedostoja, joten niitä voit editoida mieli-editorissasi omalla koneellasi ja ladata ("upload") palvelimelle.

Täytyy sanoa, että *Octave-online* vaikuttaa vähäisen käyttökokemukseni perusteella varsin käyttökelpoiselta. Sen tapa jakaa ruututila on onnistunut ja istunnon loki kuvineen kehittyy miellyttävästi niin, etteivät ikkunat peitä toisiaan. Ainoa alla olevaan yhteen ohjelmakoodiin, `SuuntakenttajaDYratk.m` vaikuttava puute näyttää olevan grafiikkaikkunaan kohdistuvan hiirisyötön toimimattomuus.

Varoitus: Lyhytkin tuumailu johtaa ilmoitukseen: "Due to inactivity, your session will expire...". Ei hätää, istuntosi jatkuu *reconnect*-painalluksella.

Octave:n "pieni kielenopas"

1. Voit kirjoittaa suoraan komentoikkunaan tyyliin: `>> komento` tai voit kirjoittaa editori-ikkunaan ja suorittaa komennot sieltä.

2. Octave:ssa operoidaan *vektoreilla* (kts. kohta 4. alla), esim.
   `>> v=[1 3 5 7]`
   Lyhyemmin: `v=1:2:7` tai `v=linspace(1,7,4)`

3. `>> help nimi` avustaa funktion `nimi` käyttöä. Toimii myös itse kirjoitettuihin funktioihin näyttämällä kaikki alkukommenttirivien tekstit.

4. **Matriisi** on suorakulmion muotoinen lukutaulukko, seuraavassa ei suoriteta matriiseilla laskutoimituksia. **Vektori** tarkoittaa lukujonoa, **skalaari** on toinen nimitys luvulle. **Vektorin pituus** on sen alkioiden lukumäärä, usein meillä 100 tai enemmänkin. Vektori voi olla myös "pystyasennossa", jolloin se voidaan mieltää yksisarakkeiseksi matriisiksi.

5. **Laskutoimitukset**: Matematiikassa (lineaarialgebrassa) määritellään samanpituisille vektoreille yhteen- ja vähennyslasku **vastinalkioittain** sekä skalaarilla kertominen kertomalla kukin alkio ko. skalaarilla. Vastinalkioittain tapahtuvalle kerto- ja jakolaskulle ei ole lineaarialgebrassa omaa merkintää.

   Vektoriohjelmointikielissä, kuten APL, MATLAB, OCTAVE, JULIA, SCILAB operoidaan vektorilausekkeilla, joilla laskettaessa tarvitaan kaiken aikaa sekä vastinalkioittain tapahtuvaa aritmetiikkaa että matriisialgebraa. Edellisessä laitetaan operaation merkin eteen piste (.) ("pisteittäin operointi") näin: `.*` `.^` `./`

   Pisteettömät operaatiot viittaavat lineaarialgebran ytimeen kuuluviin matriisilaskutoimituksiin, joita ei tässä kirjoituksessa kuitenkaan tarvita. Esim:

   ```
   >> u=[1:4]     % u = [1 2 3 4]
   >> v=[4:-1:1] % v = [4 3 2 1]
   >> w=u.*v      % w = [4 6 6 4]
   >>  u*v %error:operator*:nonconf.args
   ```

   Huomaa, että "pisteoperaatioissa" operandien on oltava samankokoisia vektoreita (yleisemmin matriiseja), tai toisen operandin on oltava skalaari.

6. **Grafiikka**: Esim: Piirrä $\sin(x)$ ja $x\sin(x)$ välillä $[-\pi, \pi]$.

   ```
   >> x=linspace(-pi,pi,100);
   >> y=sin(x);plot(x,y)
   >> hold on % Seuraava samaan kuvaan.
   >> plot(x,x.*sin(x))
   ```

   `plot(x,y)` piirtää murtoviivan $(x_1, y_1), (x_2, y_2), \ldots$ **Huomaa**, että `x.*sin(x)` laskee vastinalkioittaiset tulot: $(x_1 \sin(x_1), x_2 \sin(x_2), \ldots, x_{100} \sin(x_{100}))$.

7. **Skriptit**: Tekstitiedostoon `skripti.m` talletetaan OCTAVE-komentoja. Tiedoston komennot suorittuvat kirjoittamalla komentoikkunaan `>> skripti`

8. **Funktiot**: Tekstitiedostoon `fun.m` talletetaan komentoja, kuten skriptiin, mutta otsikkorivi on tyyppiä `function [out1,out2] = fun(in1,in2,in3)`. Funktiota kutsutaan komentoriviltä tyyliin:
   `>> [out1,out2] = fun(in1,in2,in3)`
   Syöteargumenteilla (tässä `in1,in2,in3`) tulee ennen kutsua olla arvot, joista ainakin osa on yleensä vektoreita. Sekä syöte- että tulosargumentteja voi olla kuinka monta tahansa.

9. **Funktiokahva** ("Function handle"): Kätevä tapa määritellä yhden rivin funktio tyyliin
   `f=@(x) x.*sin(x)`
   Ajattele: "at x", f saa arvon `x.*sin(x)`.
   Matematiikassa: $f : x \mapsto x \sin x$.

10. **"Vektoriäly"**: Jos `x` on vektori, niin `y=f(x)` on samanpituinen vektori, jolla `y(k)=f(x(k))` jokaisella indeksillä `k`. Tämä pätee kaikilla matemaattisilla funktioilla `f`, omat funktiot tulee kirjoittaa niin, että tämä "vektoriäly" toteutuu, mikä hoidetaan juuri pisteittäisillä laskutoimituksilla, kuten edellä.

11. **Lopeta** komento **puolipisteeseen** (;) aina, jos on odotettavissa pitkä tulostus; puolipiste estää sen. Muuttujan arvon saat näkyviin kirjoittamalla sen nimen ilman puolipistettä.

Voit myös kääntyä Googlen tai Wikipedian puoleen kysymällä vaikkapa `help matlab`, `help octave plot`, ... Lisää ohjeita on kirjoituksen lopun viitteissä.

Lähtekäämme ennakkoluulottomasti matkaan näillä eväillä ja esimerkkikoodien kommenteilla varustautumalla.

Esimerkkikoodit

https://matematiikkalehtisolmu.fi/2021/1/skriptit/
Tästä viitteestä pääset lataamaan m-tiedostot, siis tekstissä käytettävät skripti- ja funktiotiedostot. Huomaa, että m-tiedostot ovat tavallisia tekstitiedostoja, joihin voit OCTAVE:n editorin lisäksi kirjoittaa millä tahansa tekstieditorilla.

Mikä on differentiaaliyhtälö

Differentiaaliyhtälöllä tarkoitetaan yhtälöä, jossa esiintyy tuntematon funktio (merkitään y:llä tai $y(t)$:llä) ja sen derivaattoja sekä tunnettuja funktioita $a(t), b(t)$ jne. Merkitään riippumatonta muuttujaa t:llä, joka usein mielletään ajaksi. Ratkaistavana on *funktio* $y(t)$ eikä pelkkä luku.

Aivan kuten lukujen maailmassa, nytkin voidaan muodostaa yhtälöryhmiä, joissa on useampi ratkaistava funktio. Kirjoituksen (mahdollisessa) jatko-osassa käsittelen näitä.

Yhtälön *kertaluku* tarkoittaa korkeimman esiintyvän derivaatan kertalukua.

Esimerkkejä:

(1) $y' = t$ (2) $y' = y$
(3) $y' = t + y$ (4) $y' = y^2 - t$
(5) $ay'' + by' + cy = 0$ (6) $y'' - ty = 0$

Yhtälössä (1) kysytään funktiota, jonka derivaatta on funktio $f(t) = t$, no sellainenhan on $y(t) = \frac{t^2}{2}$. Siis takaperoista derivointia, eli haetaan "antiderivaattaa". Koska vakion derivaatta $= 0$, niin kaikki muotoa $\frac{t^2}{2} + c$ olevat kelpaavat, kun c on mielivaltainen vakio. (2):ssa etsitään funktiota, joka yhtyy omaan derivaattaansa. Mikähän sellainen on?

Yhtälöissä (3) ja (4) oikea puoli riippuu sekä t:stä että y:stä. Yhtälön (4) y^2 tekee siitä "epälineaarisen". Se osoittautuukin vaikeammaksi tapaukseksi kuin yksinkertainen ulkoasu antaisi olettaa.

Yhtälöt (5) ja (6) ovat toisen kertaluvun **lineaarisia** yhtälöitä. Edellinen on vakiokertoiminen, jälkimmäisessä on ei-vakiokerroin t. Kyseessä on ns. *Airyn* differentiaaliyhtälö, jota meillä on myös tilaisuus lyhyesti ihmetellä.

Koko delfiinijuoni sisältää pelkkiä ensimmäisen kertaluvun yhtälöitä, suoritamme vain pari pientä syrjähyppyä toisen kertaluvun maailmaan. Tällä kertaa ei siis käsitellä varsinaisia "peto-saalis"-malleja.

Ensimmäisen kertaluvun yhtälö:

$$y' = f(t, y) \tag{1}$$

Ratkaisu tarkoittaa jatkuvasti derivoituvaa funktiota $y(t)$, joka toteuttaa yhtälön $y'(t) = f(t, y(t))$ jollain reaaliakselin välillä $a < t < b$.

Alkuarvotehtävällä tarkoitetaan sellaisen ratkaisun etsimistä, joka toteuttaa **alkuehdon** $y(t_0) = y_0$.

Kun suureen kasvunopeus on verrannollinen suureen suuruuteen

Lähtökohtana olkoon "kaikkien differentiaaliyhtälöiden äiti":

$$y' = \alpha\, y. \tag{2}$$

Ratkaisu tarkoittaa siis funktiota y, joka toteuttaa yhtälön $y'(t) = \alpha\, y(t)$ jollain muuttujan välillä $a < t < b$.

Yhtälö kuvaa ilmiöitä, joissa funktion kasvunopeus on verrannollinen funktion arvoon kullakin ajanhetkellä t. Tällaisia ilmiöitä ovat esimerkiksi:

(a) Pankkitilin saldo, kun korko lisätään pääomaan jatkuvasti.

(b) Rajoituksista vapaa populaation kasvu.

(c) Radioaktiivisen aineen hajoaminen, kun kerroin $\alpha < 0$ edustaa hajoamisnopeutta massayksikköä kohti.

Tässä tapauksessa yhtälön **ratkaisemiseksi** tarvitsee vain miettiä, mikä onkaan sellainen funktio, jonka derivaatta on se itse (johan sitä mietittiin!). No eksponenttifunktiohan se sellainen on. Derivoimalla näet välittömästi, että kaikki muotoa

$$y(t) = C\, e^{\alpha\, t}$$

olevat funktiot, missä C on mielivaltainen vakio, toteuttavat yhtälön. Tämä on yhtälön (2) *yleinen ratkaisuparvi*. Erityisesti arvo $C = 0$ antaa vakiofunktion $y(t) \equiv 0$, joka nähdään ratkaisuksi jo suoraan yhtälöstä (2). Jos vielä vaaditaan, että ratkaisu toteuttaa *alkuehdon* $y(t_0) = y_0$, saadaan ratkaisu

$$y(t) = y_0 e^{\alpha\,(t - t_0)},$$

joka on **alkuarvotehtävän**

$$\begin{cases} y' = \alpha\, y \\ y(t_0) = y_0 \end{cases} \tag{3}$$

ratkaisu.

Esimerkki: Delfiinit Aiolian saaristossa

Tyrrhean meressä Sisilian pohjoispuolella sijaitsevan Aiolian saariryhmän vesillä arvioitiin heinäkuussa 2017 delfiinien lukumääräksi 100 yksilöä. Oletetaan, että delfiineillä oli/on rajattomasti ruokaa, ja luonnolliset saalistajat (kuten hait ym.) puuttuvat näiltä vesiltä. Oletetaan, että nettokasvunopeuskerroin (syntymät − kuolemat) on $r = 0.06$ (6 % vuodessa). Arvioi delfiinien lukumäärä tällä alueella heinäkuussa 2025.

Ratkaisu: Näillä yksinkertaisilla oletuksilla populaation kasvua kuvaa alkuarvotehtävä:

$$\begin{cases} y' = r\,y \\ y(t_0) = 100, \end{cases} \tag{4}$$

missä siis $r = 0.06, t_0 = 2017.5, y_0 = 100$. Niinpä

$$y(t) = y_0 e^{r\,(t-t_0)}$$

noilla numeerisilla arvoilla. Sittenpä käynnistetään laskentaympäristömme Octave. Koko skripti `Delfiinit1.m` on jakelupaketissamme https://matematiikkalehtisolmu.fi/2021/1/skriptit/.

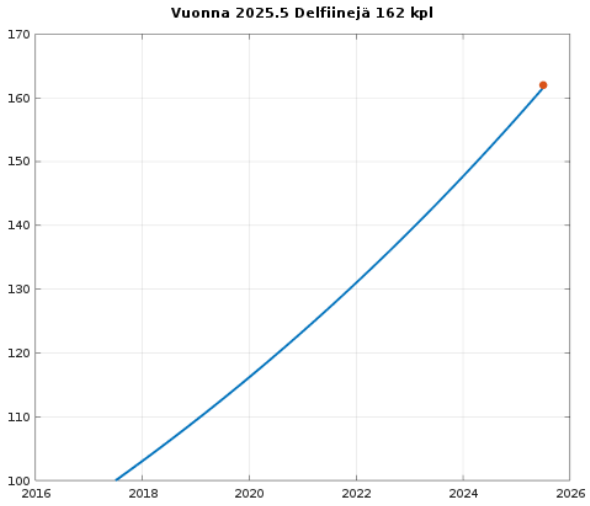
Tässä tiivistettynä tärkeimmät koodirivit:

```
r=0.06;t0=2017.5; y0=100;tloppu=2025.5;
y=@(t) y0.*exp(r.*(t-t0)); % Ratkaisufunktio
yloppu=round(y(tloppu)); % Pyöristys
% Piirretään ratkaisufunktio
t=linspace(t0,tloppu,100);
plot(t,y(t),'LineWidth',2)
hold on  % Seuraava samaan kuvaan.
% Merkitään loppupiste:
plot(tloppu,yloppu,'.','MarkerSize',10)
display('Loppuarvo vuonna 2025.5');
yloppu % Näytetään loppuarvo.
```

(Tästä tekstistä copy/paste johtaa `(')`-merkistä virheeseen.)

Komentojen selitykset:
- Rivi 1: Sijoitetaan arvot muuttujille `r,t0,y0,tloppu`
- Rivi 2: Funktiomäärittely: (Vektori)argumentin arvolla t funktio y saa arvon $y_0 e^{r\,(t-t_0)}$. (Muuttujille `y0,r,t0` on annettu kiinteät arvot ennen määrittelyä, ne eivät ole funktion argumentteja.)
- Rivi 5: Vektorille `t` annetaan arvoksi välin `[t0,tloppu]` jako sataan osaan, mikä on piirtämisen kannalta useimmiten riittävä.
- Rivi 6: `plot`-komennossa säädetään viivan leveyttä, sillä Octave:ssa saattavat oletusleveydellä näkyä hiukan himmeästi.

Saadaan kuva ja komentoikkunaan tulostus:



`Delfiinit1.m`

```
Loppuarvo vuonna 2025.5
yloppu =  162
```

Huom: Jos latasit itsellesi `Delfiinit1.m`-tiedoston, voit suorittaa sen komennot kirjoittamalla komentoikkunaan: `>> Delfiinit1` . Suositeltavaa on ottaa tiedosto editoriin: `>> edit Delfiinit1.m`, jolloin voit myös tehdä haluamiasi muutoksia ja käynnistää komennot myös editorin työkalunauhan oikean reunan ikonista tai **run**-nuolesta. Jos lataat tiedoston Octave-onlineen, se avautuu suoraan editoriin.

Jotain rajaa delfiineillekin!

Kehitellään mallia pikkuhiljaa realistisempaan suuntaan. Oletetaan, että tietty vakiomäärä B delfiinejä joutuu petokalojen, pyydysten, muskeliveneonnettomuuksien ym. uhriksi vuodessa. Tilannetta mallintaa alkuarvotehtävä:

$$\begin{cases} y' = r\,y - B \\ y(t_0) = y_0 \end{cases} \tag{5}$$

Kyseessä on *lineaarinen differentiaaliyhtälö*. Näitä esitellään yleisesti hiukan tarkemmin *analyyttisiä ratkaisuja* koskevassa kohdassa hetken päästä. Lyhyesti: Lineaarisessa yhtälössä y esiintyy ensimmäisessä potenssissa. "Homogeeniosa" on $y' = r\,y$. Epähomogeeniosassa ei y esiinny, tässä se on pelkkä B. Merkitään homogeeniosan yleistä ratkaisua y_h:lla. Lineaarisuuden takia riittää lisäksi löytää jokin koko yhtälön "erityisratkaisu" y_e. Kaikki koko yhtälön ratkaisut saadaan nyt muodossa $y(t) = C\,e^{r\,t} + y_e$.

Tämä on lineaaristen yhtälöiden yleinen ominaisuus, joka johtuu siitä, että y esiintyy ensimmäisessä potenssissa ja summan derivaatta on derivaattojen summa.

Yhtälöstä on helppo nähdä, että erityisratkaisuksi kannattaa kokeilla vakioratkaisua, koska vakion derivaatta

$= 0$. Näinpä ratkaisu suorastaan hyppää silmille:

$$y_e = \frac{B}{r}.$$

Kaikki koko yhtälön ratkaisut saadaan nyt muodossa

$$y(t) = C\,e^{r\,t} + \frac{B}{r}.$$

(Anteeksi, tämä meni hiukan pikaisesti. Jos siltä tuntuu, jätä uskon asiaksi ja jatka eteenpäin.)

Kun sijoitetaan alkuehto, saadaan ratkaistuksi kerroin C, ja ratkaisukaava voidaan kirjoittaa muotoon:

$$y(t) = (y_0 - \frac{B}{r})e^{r(t-t_0)} + \frac{B}{r}.$$

Jos oletetaan (sentään), että $r > 0$, niin tapauksessa $B < r\,y_0$ ratkaisufunktio kasvaa, kun taas päinvastaisessa tapauksessa tapahtuu päinvastoin, eli pienenee. Jos nyt sattuisi niin eriskummallisesti, että $B = r\,y_0$, niin ei liikuttaisi mihinkään, eli populaatio jatkaisi ikuisesti elämäänsä $\frac{B}{r} = y_0$ -kokoisena. Tällöin luonnollinen dynamiikka ja uhkatekijät olisivat *tasapainossa*.

Tehtävä 1. Olkoon edellä olevaan tilanteeseen liittyen $r = 0.06, t_0 = 2017.5, y_0 = 100, t_p = 2025.5$ (p: päätepiste).

(a) Täydennä ja aja skripti `Delfiinit1b.m` siinä olevien ohjeiden mukaisesti. Ainoa "koodi", joka sinun tarvitsee kirjoittaa, on yllä oleva ratkaisufunktion koodi: `y=@(t) ...`

(b) Piirrä populaatiokäyrät arvoilla $B = 0$, $B = 3$, $B = 6$, $B = 8$ ja laske loppuarvot vuonna 2025.5. Pysyykö populaatio vakiona jollain B:n arvolla, ja mitä tapahtuu suuremmilla? Voit tietysti kokeilla muitakin kuin ehdotettuja B:n arvoja.

Salaisuus: `Delfiinit1bratk.m`, mutta yritä ensin!

Seuraava kehitysvaihe on sellainen, jossa B ei ole vakio. Tällöin joudutaan tilanteisiin, joissa ratkaisukaavaa ei ole mahdollista johtaa. Elävässä elämässä kyseessä on pikemminkin sääntö kuin poikkeus. Mikä silloin neuvoksi? Lienee helppo arvata, että numeerinen ratkaisumenetelmä on avainasemassa. Kunhan työvälineitä saadaan kehitellyksi, palataan tarkempaan delfiinimallinnukseen.

Analyyttiset ratkaisumenetelmät

Esitellään joidenkin esimerkkien puitteissa tavallisimpia ratkaisumenetelmiä.

Muuttujien erottelu

Tietyntyyppisille yhtälöille voidaan suorittaa ns. *muuttujien erottelu* niin, että yhtälön ratkaiseminen palautuu kummankin muuttujan (t, y) suhteen muodostuvaan integrointitehtävään.

Esimerkki:

$$\frac{dy}{dt} = -t\,y. \tag{6}$$

Yhtälö on kirjoitettu *Leibnitzin notaatiolla*, jotta voidaan hiukan leikitellä "differentiaaleilla":

$$\frac{dy}{y} = -t\,dt.$$

Integroidaan puolittain, ts. etsitään kummallekin puolelle "antiderivaattaa". Muistellaan logaritmin derivoimiskaavaa: $\frac{d}{dy}\ln y = \frac{1}{y}$, josta seuraa:

$$\ln|y| = -\frac{t^2}{2} + C,$$

missä C on mielivaltainen vakio. Itseisarvo siksi, että myös $\frac{d}{dy}\ln(-y) = \frac{1}{y}$. Kun yhtälöön sovelletaan exp-funktiota, saadaan:

$$|y| = e^C e^{-\frac{t^2}{2}},$$

josta saadaan

$$y(t) = C\,e^{-\frac{t^2}{2}},$$

missä C:llä merkitään nyt mielivaltaista (ei pelkästään positiivisia arvoja saavaa) vakiota. (Yhtälöstä (6) nähdään suoraan, että $C = 0$ käy myös, onhan vakion derivaatta $= 0$.)

Lukija voi saada jonkinlaisia kylmiä väristyksiä tällaisesta "differentiaaleilla" dt ja dy suoritettavasta algebrasta. Itse asiassa koko homma voidaan kirjoittaa käyttämällä $\ln y(t)$:n derivoimiskaavaa (logaritmi ja yhdistetty funktio), mutta yllä esitetty on tavanomainen formaali laskutyyli näissä yhteyksissä.

Sama menettely toimii yleisemmin muotoa $y' = \alpha(t)y$ olevaan yhtälöön:

$$\int \frac{dy}{y} = \int \alpha(t)\,dt,$$

jolloin tehtävä on yhtä vaikea kuin funktion $\alpha(t)$ integrointi.

Yleisin muoto muuttujien erottelulle, joka myös motivoi tuon nimityksen, on sellainen, jossa oikean puolen funktio $f(t, y)$ on muotoa $g(t)h(y)$, eli

$$\frac{dy}{dt} = g(t)h(y),$$

josta saadaan:

$$\int \frac{dy}{h(y)} = \int g(t)dt.$$

Jos integrointi onnistuu, tästä saadaan yleensä implisiittinen (ratkaisemattomassa muodossa oleva) yhtälö y:n ja t:n välille.

Lineaarinen yhtälö yleisesti

Tällä kohden puhutaan myös lyhyesti korkeamman kertaluvun yhtälöistä.

Yhtälöt, joissa tuntematon funktio y ja sen derivaatat esiintyvät vain ensimmäisessä potenssissa ja tyyppiä $y\,y'$ olevia tuloja ei esiinny, ovat **lineaarisia**. Yleinen tilanne konkretisoituu kenties parhaiten 2. kertaluvun lineaarisen yhtälön tapauksessa:

$$a(t)\,y'' + b(t)\,y' + c(t)\,y = s(t). \qquad (7)$$

Jos oikea puoli $s(t) \equiv 0$, puhutaan *homogeenisesta lineaarisesta* yhtälöstä.

Jos erityisesti kerroinfunktiot $a(t), b(t), c(t)$ ovat vakioita, kyseessä on harmonisia värähtelyjä, sähkövirtapiirejä (LRC-piirejä) ja monia muita mallintava yhtälö. Mikä hauskinta, sen ratkaiseminen palautuu toisen asteen yhtälön juurien määräämiseen, ja ominaisuudet määräytyvät juurien ominaisuuksista.

Toisen asteen yhtälöön johdutaan sijoittamalla differentiaaliyhtälöön "yrite" $y(t) = e^{r\,t}$. Näin saadaan parametrin r määräämiseksi yhtälö

$$a\,r^2 + b\,r + c = 0.$$

Taas tarvitaan siis vain eksponenttifunktion derivoimiskaavaa (ja toisen asteen yhtälön ratkaisukaavaa).

Lineaarisille (tässä toisen kertaluvun) yhtälöille pätee yleisesti:

(a) Homogeeniosan (oikea puoli = 0) yleinen ratkaisu on muotoa $y_h(t) = C_1\,y_1(t) + C_2\,y_2(t)$, missä y_1 ja y_2 ovat homogeeniyhtälön ratkaisuja, joiden suhde ei ole vakio. (Puhutaan lineaarisesti riippumattomista ratkaisuista.)

(b) Koko epähomogeenisen yhtälön yleinen ratkaisu on $y(t) = y_h(t) + y_k(t)$, missä y_k on jokin koko epähomogeenisen yhtälön ratkaisu, puhutaan "erityisratkaisusta".

Nämä seuraavat suoraan lineaarisuudesta, koska myös derivaatat käyttäytyvät lineaarisesti (summan derivaatta on derivaattojen summa).

Näitä periaatteita käytettiin edellä anteeksipyydellen 1. kertaluvun lineaarisen yhtälön tapauksessa.

Kvalitatiiviset menetelmät

Kvalitatiivinen menetelmä tarkoittaa lyhyesti sanottuna pyrkimystä saada tietoa ratkaisujen laadullisesta käyttäytymisestä suoraan yhtälöstä. Tietokonegrafiikan ja tehokkaan laskennan mahdollisuudet ovat edistäneet näitä pyrkimyksiä ratkaisevasti ja johtaneet aihepiiriin nimeltään "dynaamiset systeemit". Tässä tulevat näkyviin "(epä)stabiilisuus", "perhosefekti", "kaaos", pitkän aikavälin ilmiöt. Viite [HW] on erinomainen lähde, mukaansatempaavaan tyyliin kirjoitettu.

Suuntakenttä

Kun lasketaan yhtälön $y' = f(t, y)$ tilanteessa funktion f arvoja (t, y)-tason pisteissä, saadaan kokoelma ratkaisufunktion derivaatan arvoja.

Muodostamalla (t, y)-tasoon pistehila ja piirtämällä kuhunkin hilapisteeseen (t_k, y_k) lyhyt jana arvon $f(t_k, y_k)$ ilmoittamaan ratkaisukäyrän tangentin suuntaan, saadaan viivakuva (vektorikenttä), jota kutsutaan differentiaaliyhtälön **suuntakentäksi**.

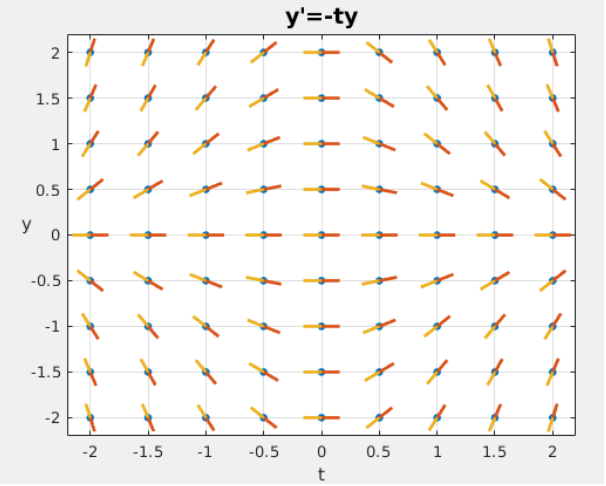
Edellä ratkaistiin muuttujien erottelulla yhtälö $y' = -t\,y$. Katsotaanpa, miltä suuntakenttä näyttää.

Ratkaisua katsomatta nähdään suoraan yhtälöstä:

- Ratkaisu on vaakasuora t- ja y-akselin pisteissä.
- Kiinteällä t:n arvolla ($t \neq 0$) kulmakerroin jyrkkenee, kun $|y|$ kasvaa. Samoin käy kiinteällä y:n arvolla, kun $|t|$ kasvaa.
- Kenttä on symmetrinen O:n suhteen ja antisymmetrinen akselien suhteen.

Harjoituksen vuoksi voit hahmotella ruutupaperille.

Tässä on Octave:lla (Matlab:lla) piirretty kuva, jossa selvyyden vuoksi on otettu harva pistehila.



`suuntakentta1.m`

Suuntakentän piirtäminen

Octave-jakelussamme
https://matematiikkalehtisolmu.fi/2021/1/skriptit/
on `suuntakentta1.m`-tiedosto, jonka voit ladata työhakemistoosi ja suorittaa komennolla `>> suuntakentta1`. Kannattaa avata editorissa: `>> edit suuntakentta1.m`. (Jos lataat tiedoston "Octave-onlineen", se avautuu suoraan editoriin.) "Markkerien" koko näkyy erilaisena eri versioissa, kokeile esim. `'MarkerSize',n` välillä $5 \leq n \leq 15$. Muuten voit muokata vaikkapa muuttamalla hilaväliä ja aluetta.

Differentiaaliyhtälöä voit vaihtaa muuttamalla funktion $f(t, y)$ määrittelyä. Funktiomäärittelyn syntaksi on: `f = @(t,y) -t.*y`.
Kertaukseksi, ajattele: "at (t, y), arvo $-t\,y$", matematiikassa: $f : (t, y) \mapsto -t\,y$.

Huom: Funktiota sovelletaan hilapisteistöön, jonka tuottaa alla oleva `meshgrid`-komento, joka muodostaa kaksi *samankokoista* matriisia `t` ja `y`, edellinen sisältää pisteistön t- ja jälkimmäinen y-koordinaatit.

Pisteittäisten laskutoimitusten käyttö ja notaatio on omaksuttu Matlab:n perässä Octave:n lisäksi muihin vektori/matriisikieliin, kuten Julia, Scilab.

Kaikissa muissakin mainituissa on Matlab:sta peräisin olevat voimakkaat vektorikenttien piirtotyökalut, jotka tässä yhteydessä ovat näillä riveillä:

```
% Hilapisteistön koordinaattimatriisit(t,y):
[t,y]=meshgrid(tpoints,ypoints);
% Piirretään hilapisteisiin viivakenttä:
quiver(t,y,pt,py,viivaskaala);
quiver(t,y,-pt,-py,viivaskaala); % Viiva ...
% ..taaksepäin. Kommentoi pois, jos et halua.
```

Jos kiinnostaa kokeilla mieliohjelmistollasi suuntakenttiä, voit hakea Googlella vaikka hakusanoilla *direction field, slope field*. Esim. *Geogebralle* on tehty: https://www.geogebra.org/m/W7dAdgqc "Slope field plotter", varsin siisti.

Octave-ympäristössä on käsissämme työkalut, joita voidaan muunnella halumme mukaan, lisätä eri menetelmin laskemiamme ratkaisukäyriä valituilla alkuarvoilla, laskea ratkaisujen arvoja halutuissa pisteissä, tarkastella virhekäytöstä eri menetelmien välillä jne. Samalla nähdään, miten pienellä määrällä koodirivejä saadaan varsin vaativiakin tehtäviä suoritetuksi.

Ratkaisukäyriä suuntakenttään

Tarkasteltava yhtälö $y' = -t\,y$ ratkaistiin yllä, joten voidaan katsoa, miten suuntakenttä ja oikeat ratkaisukäyrät sopeutuvat yhteen. Edellä saatiin muuttujien erottelulla alkuarvotehtävälle ratkaisu

$$y(t) = y_0 e^{-\frac{t^2}{2}},$$

kun $y_0 = y(0)$.

Voidaan piirtää suuntakenttäkuvaan vaikka ratkaisut alkuarvoilla $y(0) = 0.5, y(0) = 1, y(0) = 1.5$ komennoilla:

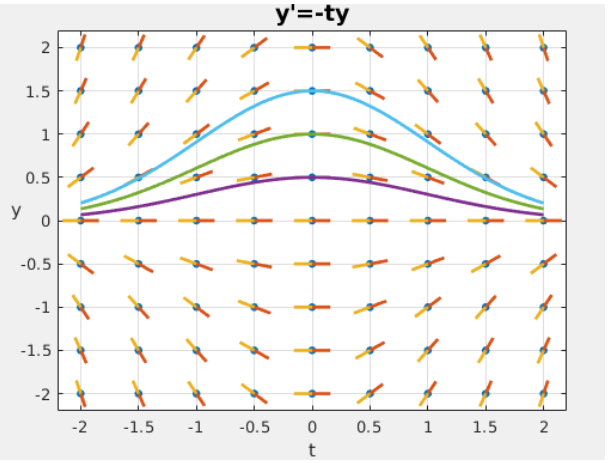
`yf=@(y0,t)y0.*exp(-0.5.*t.^2)` % `t.^2`, kts. alla.

```
hold on
t=linspace(-2,2,100);
plot(t,yf(0.5,t),'LineWidth',2)
plot(t,yf(1,t),'LineWidth',2)
plot(t,yf(1.5,t),'LineWidth',2)
```

("copy/paste" komentoriville toimii, mutta merkkijonojen 'hipsukat' koodautuvat väärin.)

Huomaa: `t.^2`, koska funktiomäärittelyssä `t`:n pitää voida olla vektori, kuten käyttöesimerkeissä heti nähdään. Kertolaskuissa riittäisi tässä `*`, koska toinen tekijä on skalaari, mutta johdonmukaisuuden nimissä käyttäkäämme pistettä. (Miksi `t^2` ei kelpaa, vaikka 2 on skalaari? Koska `t^2` tarkoittaa (matriisi)tuloa `t*t`.)

Itse asiassa lisäsin nämä rivit `suuntakentta1.m`-skriptiin, kuten näet kirjoittamalla Octave:n komentoikkunaan: `>> suuntakentta1`. Kokeile ja piirrä lisää, jos huvittaa, tai jos ei huvita, katso vain herkeämättä kuvaa.



Suuntakenttä ja ratkaisuja

Milloin voidaan ratkaista analyyttisesti?

Edellä esiteltiin joitakin ratkaisutekniikoita. Kerrataan vielä: Puhumme *symbolisesta*, *analyyttisesta* tai *suljetussa muodossa olevasta* ratkaisusta, jos se voidaan esittää tunnettujen alkeisfunktioiden avulla.

Esimerkki:

$$y' = y^2 - t.$$

Yhtälö näyttää varsin vaatimattomalta, toki siinä on y^2, joten se on *epälineaarinen*, mutta silti siistin näköinen. Vaan eikö mitä, tarvittiin kuuluisa matemaatikko *Henri Poincaré*, joka osoitti n. 100 vuotta sitten, että yhtälöllä ei ole ratkaisukaavaa tavallisten matemaattisten alkeisfunktioiden avulla eikä edes niiden integraalien avulla lausuttuna. Tämän todistaminen vaati syvällistä matemaattista koneistoa, ns. *Galois'n* teoriaa.

Kehittyneimmät tietokonealgebraohjelmistot sisältävät laajan valikoiman tekniikoita alkaen edellä esitetyistä perustekniikoista varsin kehittyneisiin matemaattisiin menetelmiin.

Tällä syntaksilla toimii MATLAB/OCTAVE:n *symbolic toolbox*'n ratkaisija.

```
>> syms t y(t)
>> dsolve(diff(y(t),t)==y(t)^2-t)
```

OCTAVE-online ei selviä tehtävästä, MATLAB palauttaa "tolkuttoman" kaavan, jossa esiintyy murtolukuindeksein varustettuja kahden lajin *"Besselin funktioita"*.

Symbolilaskentaan erikoistunut MAPLE-ohjelmisto toimii siistimmin:

```
> ratk:=dsolve(diff(y(t),t)=y(t)^2-t);
> latex(ratk);
```

Kun tulos tulkitaan LaTeX-ohjelmalla (kuten koko tämä kirjoituskin), saadaan:

$$y\left(t\right)=-\frac{C\,\mathrm{Ai}^{(1)}\left(t\right)+\mathrm{Bi}^{(1)}\left(t\right)}{C\,\mathrm{Ai}\left(t\right)+\mathrm{Bi}\left(t\right)},$$

missä osoittajan yläindeksit tarkoittavat derivaattoja. (Kun funktiot tunnetaan, saadaan derivaatat suoraan diffyhtälöstä.)

Onko tulos ristiriidassa *Poincaré*:n todistuksen kanssa? No ei, vaan "suljettu muoto" voidaan yleistää sallimalla erinäisiä ns. "erikoisfunktioita", joista yllä mainitut Besselit ja Airyt ovat esimerkkejä. MAPLE:a voidaan pyytää kertomaan, mitä menetelmiä se yrittää. Saadaan pitkä lista, alkaen lineaarisista ja muuttujien erottelusta, päätyen tällaiseen:

```
trying Riccati Special (erikoisfunktiot)
Riccati Special successful
```

Tässä tapauksessa saatiin siisti kaava, jonka laskeminen on aivan yhtä helppoa kuin jos Airyjen sijalla olisi vaikka sinit ja kosinit. Laskentateho riippuu Airyfunktioiden laskenta-algoritmista.

Kuten sanottu, "suljetun muodon" käsite on erikoisfunktioiden avulla laajennettavissa perinteisestä huomattavassa määrin, ja se laajenee sitä mukaa kuin uusia erikoisfunktioita otetaan käyttöön ja niille kehitetään luotettavia numeerisia algoritmeja, jotka myös standardisoidaan niin, että eri ohjelmointikielet ja ohjelmistot ja jopa laskimet ne tuntevat yksiselitteisesti. (Tosin ajan mittaan laskimet voivat tällä menolla kasvaa "Stadionin tornin korkuisiksi".)

Yleistettyjen ratkaisumuotojen määrittäminen ei enää oikein ole ihmisen käsissä, vaan "matemaattista tekoälyä" edustaviin tietokonealgebraohjelmistoihin on hyvä turvautua analyyttisten ratkaisujen alueella ainakin silloin, kun perustekniikat eivät pure. Toki tuloksen pätevyyttä kannattaa aina tutkia ja erityisesti, jos ratkaisukaava on toivottoman mutkikas, jolloin sen käyttökelpoisuuskin voi olla kyseenalainen.

Pohdinta: Airyn funktioilla on monta eri luonnehdintaa. Esimerkiksi ne ovat alkuesittelyssä mainitun "Airyn differentiaaliyhtälön" $y''-ty=0$ lineaarisesti riippumattomia ratkaisuja (niiden suhde ei ole vakio). Eräs laskenta-algoritmi olisi tuon differentiaaliyhtälön numeerinen ratkaiseminen. Siis ratkaisemalla Airyn differentiaaliyhtälö numeerisesti saadaan differentiaaliyhtälön $y'=y^2-t$ (yleistetty) analyyttinen ratkaisu. Miksi emme yhtä hyvin ratkaisisi yhtälöämme suoraan numeerisesti? Toki näin siistin kaavan käyttö on helpompaa, kun/jos Airyn yhtälön ratkaisu on saatavilla luotettavasti ohjelmoituna. Mutta monessa tapauksessa suora numeerinen ratkaisu on helpompi, tehokkaampi ja ennen kaikkea **yleispätevämpi** tie.

Entä tämä ajatus: Jos ratkaistavana olisi (lineaarinen) yhtälö $y''=-y$, niin hetihän näkisimme derivoimalla 2 kertaa, että $\cos t$ ja $\sin t$ toteuttavat yhtälön ja ovat lineaarisesti riippumattomat, onhan $\frac{\sin t}{\cos t}=\tan t\neq$ vakio, joten yleinen ratkaisu on $y(t)=C_1\cos t+C_2\sin t$. Mistä tiedämme, millä algoritmilla meille kosini- ja sini-näppäimet laskevat? Kukaties ne ratkaisevat numeerisesti yhtälön $y''=-y$ alkuehdoilla $y(0)=1$ ja $y(0)=0$.

Miten edellä saatua ratkaisukaavaa käytettäisiin?

Jos jatkaisimme MAPLE:lla, ei olisi mitään ongelmaa. Mutta kirjoituksen henki on sellainen, että haluamme pitäytyä vapaasti saatavassa OCTAVE:ssa (tuota annettua MAPLE-kaavanvääntöä lukuunottamatta). Pitää siis selvittää, tunteeko OCTAVE nuo ratkaisukaavassa esiintyvät funktiot. Kirjoitin tätä varten skriptin `AnalratkAiry.m`, jossa asia on selvitetty ja jolla pääset piirtämään ja laskemaan ratkaisuja

Tehtävä: Piirrä yhtälömme suuntakenttä ja siihen joitakin ratkaisukäyriä `AnalratkAiry.m`-skriptin avulla samaan tapaan kuin edellä `suuntakentta1.m`-skriptissä.

Käytännön mallinnustehtävissä on joka tapauksessa tässä laajennetussakin merkityksessä suljetun muodon ratkaisun puuttuminen enemmän sääntö kuin poikkeus myös symboliohjelmistoja käytettäessä.

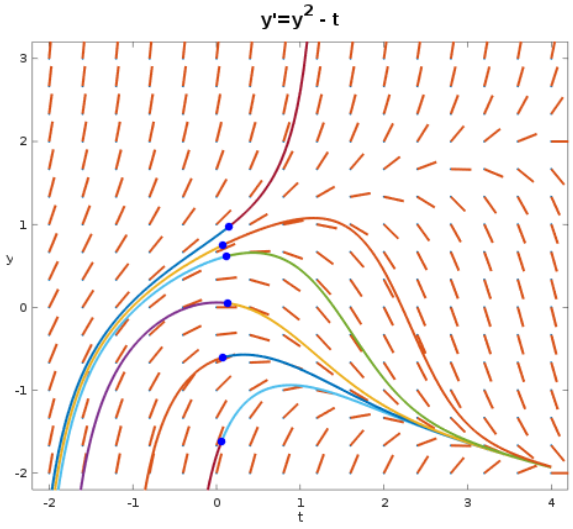
Ratkaisukäyriä vuorovaikutteisesti suuntakenttään

Jatketaan esimerkkimme parissa noudattaen yleistyskelpoista linjaa, eli mennään hiukan asioiden edelle ja käytetään MATLAB/OCTAVE:n **numeerista** ratkaisijaa `ode45` (`ode` on lyhennys termille "ordinary differential equation").

Olisipa hauskaa, jos voitaisiin piirtää suuntakentästä hiirellä valituista pisteistä lähteviä ratkaisukäyriä.

Kirjoitin skriptitiedoston `SuuntakenttajaDYratk.m`, jossa käytetään OCTAVE:n `ginput`-funktiota alkupisteen poimimiseen suuntakenttäkuvasta, ja valitusta pisteestä lähtevä ratkaisuapproksimaatio lasketaan tuolla "mustalla laatikolla" `ode45`. (Valitettavasti `ginput` ei näytä toimivan `Octave-online`:ssa.)

Tässä on tulos, josta näkyy, mitä kohtia kuvassa olen klikkaillut.



`SuuntakenttajaDYratk.m`

Kahden ylimmän alkupisteen paikkeilla nähdään, että hyvin pieni alkuarvon muutos johtaa täysin eriluonteiseen ratkaisukäyrään. Neljä alinta sijaitsevat varsin kaukana toisistaan, mutta johtavat samaan "suppiloon" puristumiseen ajan kasvaessa. Edellisessä näkyy **epästabiili käytös**, jossa voi nähdä kuuluisan "perhosefektin" ilmentymän. Jälkimmäinen taas kertoo vahvasta **stabiilisuudesta**.

Mitäs nämä numeerisella algoritmilla lasketut käppyrät ovat? Todistavatko ne ratkaisujen olemassaolon?

Tärkeää ymmärtää:

- Differentiaaliyhtälön ratkaisun olemassaolo ei edellytä *ratkaisukaavan* olemassaoloa.
- Numeerisella algoritmilla lasketaan approksimaatio oletetulle ratkaisulle, mutta se ei todista, että ratkaisu on olemassa.
- Kvalitatiivisten menetelmien avulla voidaan saada (oletetun) ratkaisun käyttäytymisestä ymmärrys tarkastelualueen eri osissa nojautumatta ratkaisukaavaan, jota ei siis välttämättä ole.
- Ratkaisun olemassaolon selvittämiseen on yleisiä ehtoja, josta seuraavassa.

Ratkaisun olemassaolo ja yksikäsitteisyys

Kyseessä on siis ratkaisu alkuarvotehtävälle

$$\begin{cases} y' = f(t,y) \\ y(t_0) = y_0 \end{cases} \tag{8}$$

Ajatellaan ihan aluksi suuntakenttää. Koska $f(t,y)$:llä on yksikäsitteinen arvo jokaisessa (t,y)-pisteessä, eivät suuntajanat voi leikata toisiaan, ja siis ratkaisukäyrät eivät leikkaa. Eikös tämä jo ole yksikäsitteisyyttä? Mutta voisivathan ne sivuta toisiaan, kuten näyttäisi voivan käydä jossain tulevaisuudessa, kun katsotaan vaikkapa kuvaa $y' = y^2 - t$. Kuitenkin tällainen "suppilomainen" käytös on useimmiten tyyppiä $e^{\alpha t}$, missä $\alpha < 0$, jolloin sivuaminen tapahtuu vasta "äärettömän kaukana".

Katsotaanpa esimerkkiä, jossa sivuaminen tapahtuukin äärellisesä ajassa.

Esimerkki: Vuotava ämpäri

Ajatellaan, että pihalle on unohdettu vedellä täytetty ämpäri, jonka pohjassa on reikä. Kun myöhemmin havaitaan, että ämpäri on tyhjä, voidaanko (asiaankuuluvat parametrit tuntemalla) laskea, milloin se oli täysi. No eipä tietenkään! (Vrt. radioaktiiviseen hajoamiseen perustuva iänmääritys, jossa päästään käsiksi kaukaiseen menneisyyteen.)

Jos vedenpinnan korkeutta merkitään $y(t)$:llä, voidaan johtaa yhtälö

$$y' = -\alpha\sqrt{y},$$

missä $\alpha = \sqrt{2g}\frac{a}{A}$, a on reiän halkaisija ja A lieriön pohjan halkaisija (kts. [HW] s.160).

Muuttujien erottelulla saadaan:

$$\int \frac{dy}{\sqrt{y}} = -\alpha \int dt + C.$$

Kun ajatellaan käänteisesti potenssin derivoimiskaavaa, saadaan:

$$2\sqrt{y} = -\alpha\, t + C \implies y(t) = \frac{1}{4}(C - \alpha\, t)^2.$$

(Korkeus $y(t)$ ei voi saada negatiivisia arvoja, kuten yhtälöstäkin näkyy.) Suoraan yhtälöstä näkyy vakioratkaisu $y(t) \equiv 0$.

Valitaan yksiköt niin, että $\alpha = 1$ ja lieriön tilavuus $= 1$, lieriö olkoon täysi hetkellä t_0, ts. $y(t_0) = 1$ ja olkoon t_1 seuraava hetki, jolloin se on juuri tyhjentynyt, $y(t_1) = 0$.

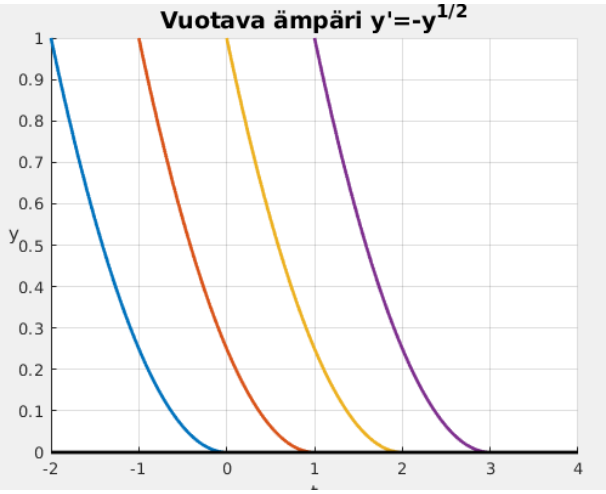
$$1 = y(t_0) = \frac{1}{4}(C - t_0)^2 \implies C = t_0 + 2 = t_1.$$

($t_0 - 2$ ei käy, kun on oltava $t_1 > t_0$.)

Ratkaisuksi saadaan:

$$y(t) = \begin{cases} \frac{1}{4}(t - t_1)^2, \text{ kun } t_0 \le t \le t_1 \\ 0, \text{ kun } t > t_1. \end{cases}$$

(Kun ämpäri on tyhjentynyt, se pysyy tyhjänä.)



Skripti `ampari.m`

Kyseessä on jatkuvasti derivoituva funktio myös "liitoskohdassa" $t = t_1$, sillä vasemman- ja oikeanpuoleiset derivaatat ovat $= 0$, ja differentiaaliyhtälö toteutuu niin vasemmalla kuin oikealla. Nähdään, että jos alkuarvoksi otetaan $y(t_0) = 0$, niin tämän alkuehdon toteuttavia ratkaisuja ovat myös kaikki ehdon $y(t_a) = 0$ toteuttavat, joilla $t_a < t_0$. Kuvasta konkreettisesti: Oikeanpuoleinen ratkaisukäyrä toteuttaa alkuehdon $y(3) = 0$, mutta niin toteuttavat myös kaikki sen vasemmalla puolella olevat.

Jos sen sijaan alkuehdoksi otetaan $y(t_0) = y_0$, missä $y_0 > 0$, ei mikään muu ratkaisukäyrä kulje tuon pisteen kautta.

Toisin sanoen: **Jos ämpärissä on vähänkin vettä jäljellä, voidaan laskemalla selvittää hetki, jolloin se oli täysi.**

Ratkaisujen olemassaololle ja yksikäsitteisyydelle on yleisiä, funktion $f(t, y)$ ominaisuuksiin liittyviä ehtoja.

Tarvitaan *osittaisderivaatan* käsitettä. Se tarkoittaa yksinkertaisesti kahden muuttujan funktion tapauksessa sitä, että derivoidaan vain toisen muuttujan suhteen pitäen toista vakiona (eikä anneta juhlallisen näköisen merkinnän hämätä.)

Esim:

$$\frac{\partial}{\partial t}(t + y^2) = 1, \ \frac{\partial}{\partial y}(t + y^2) = 2y, \ \frac{\partial}{\partial y}(ty^2) = 2ty.$$

Lause 1. Tarkastellaan alkuarvotehtävää (8). Oletetaan, että f ja $\frac{\partial f}{\partial y}$ ovat jatkuvia alkuarvopisteen (t_0, y_0) sisältävässä suorakulmiossa R: $|t - t_0| \le \alpha, |y - y_0| \le \beta$. Alkuarvotehtävällä on yksikäsitteinen ratkaisu alkuarvopisteen (t_0, y_0) ympäristössä $|t - t_0| \le \min(\alpha, \beta/M)$, missä $M = \max_R |f(t, y)|$.

Lauseesta on erilaisia variantteja. Osittaisderivaattaehtoa voidaan lieventää, puhutaan *Lipschitz*-ehdosta ([HW] s. 165, [CF] s. 669). Lineaarisille yhtälöille on tietyin edellytyksin laajemmalla alueella voimassa oleva versio jne.

Alkuperäisen todistuksen esitti ranskalainen *Emile Picard* vuonna 1893 ja tulosta tarkensi suomalainen matematiikan vaikuttajahenkilö ja "isähahmo" *Ernst Lindelöf* hiukan myöhemmin. Kts. [CF] s. 649 ja [OLehto]. Todistuksen idea on hyvin esitetty viitteessä [CF]. Kyseessä on iteratiivinen menetelmä, "Picardin iteraatio", ja se on abstrakti olemassaolotodistus. Vaihtoehtoinen konstruktiivinen todistus on esitetty viitteessä [HW], jossa yhtenä osana on numeeristen menetelmien perustana oleva *Eulerin menetelmä*, seuraava aiheemme.

Mitä lause sanoo esimerkkiyhtälöstämme $y' = -\sqrt{y}$? Siis $f(t, y) = -\sqrt{y}$, f ei riipu t:stä, jolloin osittaisderivaatta on ihan tavallinen derivaatta. Siis:

$$\frac{\partial f}{\partial y} = \frac{d}{dy}(-\sqrt{y}) = -\frac{1}{2\sqrt{y}}.$$

Derivaattaa $-\frac{\partial f}{\partial y}$ ei ole, kun $y = 0$, joten lauseen oletukset alkuarvolle $y_0 = 0$ eivät ole voimassa, mutta ovat, kun $y_0 > 0$. Niinpä esimerkkimme on sopusoinnussa lauseen kanssa.

Numeeriset ratkaisumenetelmät

Eulerin menetelmä

Jos $y' = \alpha\, y$ on kaikkien differentiaaliyhtälöiden äiti, niin *Eulerin menetelmä* on sitä kaikkien numeeristen ratkaisijoiden suhteen.

Ratkaistavana olkoon alkuarvotehtävä:

$$y' = f(t, y), \quad y(t_0) = y_0.$$

Ajatellaan suuntakenttää. Piirretään alkuarvopisteeseen (t_0, y_0) lyhyt suuntakenttäjana, jonka suunta on ratkaisukäyrän tangentin suunta: $y'(t_0) = f(t_0, y_0)$.

Suuntanuolen loppupäässä tehdään sama uudestaan aikapisteessä $t_1 = t_0 + h$, missä h on lyhyt aika-askel. Olkoon suuntajanan loppupiste (t_1, y_1), eli

$$y_1 = y_0 + hy'(t_0) = y_0 + h\,f(t_0, y_0).$$

Tässä meillä on *Eulerin menetelmän* askel, jota toistamalla saadaan pisteet $(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$, joita yhdistävä murtoviiva approksimoi ratkaisua sitä tarkemmin, mitä pienemmin askelin h kuljetaan.

Jotta saataisiin myös arvio virheelle, joka tehdään yhdellä askeleella, kun ratkaisukäyrä korvataan tangentillaan, muodostetaan *Taylorin*[1] kehitelmä pisteessä t. Jos $y(t)$ on ratkaisufunktio,

$$y(t+h) = y(t)+h\,y'(t)+R(h) = y(t)+h\,f(t, y(t))+R(h).$$

Tässä jäännöstermi $R(h)$ toteuttaa ehdon $|R(h)| \leq M\,h^2$, kun h on riittävän pieni, missä $M = \max_{|t|\leq|h|} |y''(t)|$.

Vakiota M ei etukäteen tunneta, koska y'' on tuntematon. Tärkeää on, että yhdellä askeleella syntyvä menetelmävirhe (lokaali virhe) on verrannollinen neliöön h^2. Jos askelia otetaan n kappaletta, niin $h \sim \frac{1}{n}$, joten n:n askeleen synnyttämä kokonaisvirhe on verrannollinen askeleeseen h. (Tämä on hiukan ylimalkainen päätelmä, mutta uskottava ja vastaa todellisuutta varsin hyvin.) Likipitäen pätee:

Eulerin menetelmän kokonaisvirhe puolittuu, kun askelpituus puolittuu.

Menetelmä on siten varsin tehoton ja siksi onkin kehitetty koko joukko huomattavasti tehokkaampia menetelmiä. Näissä muunnetaan Eulerin menetelmää niin, että otetaan koeaskelia sopivasti esim. puolikkaalla askelpituudella ja sitten tietyn kriteerin mukaan edetään optimaalisesti painotetun keskiarvon suuntaan. Tällainen on mm. edellä mainittu `ode45`, joka on varsin laadukas yleisratkaisija. Siinä kokonaisvirheen kertaluku on 4 verrattuna Eulerin kertalukuun 1. Menetelmän ovat kehittäneet matemaatikot: *Runge, Kutta, Fehlberg.*

Eulerin menetelmän ohjelmointi

Iterointiaskeleen muuttaminen ohjelmakoodiksi on mitä luontevinta. Katsotaan esimerkiksi yhtälöä

$$y' = t + y, \quad y(0) = 0.$$

Analyyttinen ratkaisu on helppo muodostaa. Lineaarisen yhtälön homogeeniosa antaa Ce^t:n, ja epähomogeenisen erityisratkaisu keksitään muodossa $a - t$. Vakiolle a saadaan arvo -1. Myös Octave osaa:

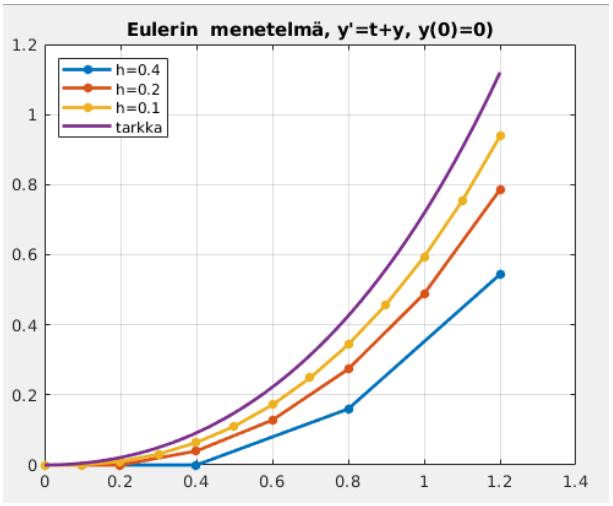
```
>> syms t y(t); dsolve(diff(y(t))==t + y)
ans = C1*exp(t) - t - 1
```

Alkuehdolla $y(0) = 0$ saadaan $C1 = 1$.

Eulerin menetelmä juoksee alla olevalla silmukalla, jolla syntyy kuvan alin murtoviiva.

```
f=@(t,y) t+y;
h=0.4; n=3; % väli: [0,1.2]
t=0:h:n*h   % t=[0,h,2*h,3*h]
y(1)=0 % Vektorin indeksointi alkaa 1:stä
for k=1:n
   y(k+1)=y(k)+h*f(t(k),y(k));
end;
```

Skriptissä `Euleresim1.m` on tämä toteutettu a), b) ja c)-kohdissa askelpituuksilla 0.4, 0.2, 0.1. Sopivin alustuksin ja plot-komennoin syntyy kuva:



`Euleresim1.m`

Nyt voidaan kirjoittaa oma funktio, jota kutsutaan, kun halutaan menetelmää soveltaa. (Tiedosto `eulerS.m`)

```
function [T,Y]=eulerS(f,Tspan,ya,n)
% Euler Skalaari, HA
% in: f     - funktiokahva("function handle")
%     Tspan - Tarkasteluväli
%     ya    - alkuarvo pisteessä Tspan(1)
%      n    - Jakovälien lkm.
% out: T    - Aikapisteet
%      Y    - Lasketut y-arvot T-pisteissä
% Esim:  y'=t+y, y(0)=1
%        f=@(t,y)t+y; close all
%        [T,Y]=eulerS(f,[0 4],1,6);
%        LW='LineWidth';MS='MarkerSize';
%        plot(T,Y,'.-',LW,2,MS,10)
a=Tspan(1);b=Tspan(2);
h=(b-a)/n;
```

[1] Taylorin kehitelmää ei tarvitse tuntea, usko vain tuo arvio!

```
Y=zeros(n+1,1);T=(a:h:b)'; % Sarakevektorit,
Y(1)=ya;                   % kuten ode45:ssä
for j=1:n
   Y(j+1)=Y(j)+h*f(T(j),Y(j));
end;
```

Huomataan, että suurin osa riveistä on kommentteja, jotka tulevat näkyviin komennolla: `>> help eulerS` . Funktion kutsu on identtinen "tehoratkaisijan" `ode45` kanssa, paitsi tässä on viimeinen parametri `n`, jonka `ode45` säätää vaadittujen virhetoleranssien perusteella.

Jos halutaan käyttää *Eulerin menetelmää* tehtävään, jonka ratkaisua ei tunneta, kuten käytännössä useimmiten on tilanne, voidaan arvioida alkuaskelpituus h ja verrata tulosta muutamaan askeleen puolittamalla saatuun. Jos tulosvektori ei vaadittavan laskentatarkkuuden puitteissa muutu, voidaan tulos hyväksyä. (Tarkempia askeleen säätämis- ja virheen arvioimismenetelmiä on luotettavimmissa ohjelmistoissa olevissa koodeissa kuten `ode45`.)

Kiintoisana lähihistorian esimerkkinä kannattaa mainita elokuvan "Hidden Figures" päähenkilön oivallus käyttää *Eulerin menetelmää* laskettaessa *John Glenn'n* avaruuslennon rataa. Pääosan esittäjä eläytyy *Katherine Johnsonin*, arvostetun mustaihoisen matemaatikon osaan. Mainittakoon, että *Katherine Johnson* kuoli vuonna 2018, 101:n vuoden iässä. Aiheesta on *Anne-Maria Ernvall-Hytösen* kirjoitus ja suositus: https://matematiikkalehtisolmu.fi/2017/hidden_figures.pdf

Esihistoriallinen näkökulma

Babylonialaisissa nuolenpääkirjoituksella (n. 3000 v. eaa.) esitetyissä tauluissa annetaan menetelmä lainanlyhennyksen korkoa korolle laskennassa, kun lyhennys tapahtuu tasaisin aikavälein, siis aivan kuten meillä nykyisinkin. Tämä on täsmälleen sama menetelmä kuin yhtälön $y' = \alpha\, y$ alkuarvotehtävän numeerinen ratkaiseminen *Eulerin menetelmällä.*

Paluu delfiinien pariin

Otsikon *Jotain rajaa delfiineillekin!* alla mallissa (5) oli vakio B edustamassa kasvua rajoittavia tekijöitä, kuten kalaverkkoja, moottoriveneonnettomuuksia jne. Realistisempi malli saadaan olettamalla tämän termin olevan ajasta riippuva. Kesäkaudella on otaksuttavasti suuremmat vaarat.

Olkoon kasvua rajoittava kerroinfunktio

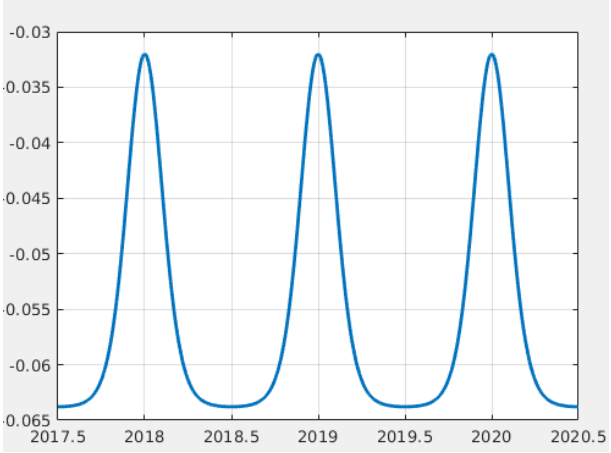
$$b(t) = -0.032(2 - \exp(-5(\sin \pi\, t)^2)),$$

missä aikaa mitataan vuosissa. Voidaan ajatella, että kokonaiskasvukerroin on ajasta riippuva: $r + b(t)$, joten differentiaaliyhtälö on

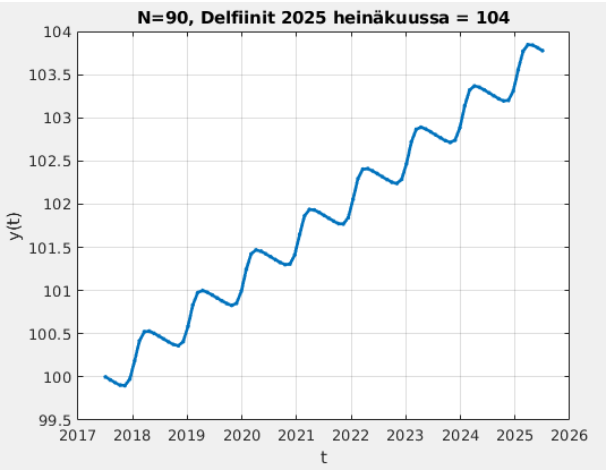
$$y' = (r + b(t))y.$$

Kasvukerroin on selvästi jaksollinen, jaksona 1 (vuosi).

```
b=@(t) -0.032*(2 - exp(-5*(sin(pi*t)).^2))
t=linspace(2017.5,2020.5,500);
plot(t,b(t),'LineWidth',2);grid on
```



Ratkaisemista ajatellen kyseessä on yhtälö $y' = \alpha(t)\, y$, joka voidaan ratkaista muuttujien erottelulla, kuten edellä esitettiin. Tällä kertaa $\alpha(t)$-funktion antiderivaatan määrittämiseen ei ole mitään toiveita. Niinpä turvaudutaan *Eulerin* menetelmään. Skriptissä `Delfiinit2.m` suoritetaan ratkaisu askelmäärillä $N = 90$ ja $N = 180$, piirretään vain edellisellä ja lasketaan loppuarvot, jotka eivät eroa toisistaan. Aja taas ja kokeile huviksesi vaikkapa pienentämällä N:n arvoa.



`Delfiinit2.m`

Parannettu delfiinimalli, vai huononnettu

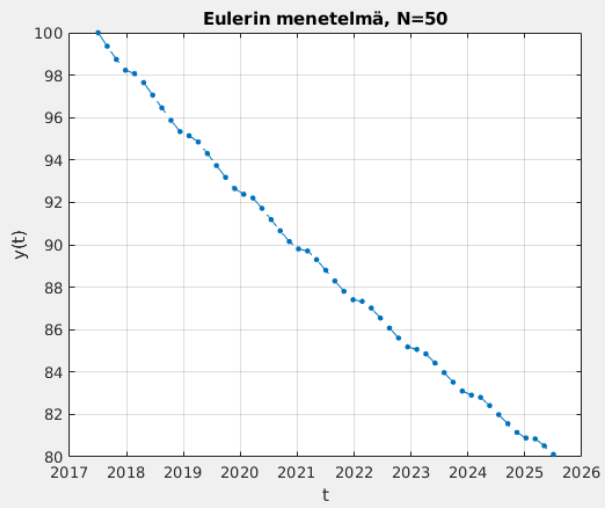
Mallinnuksessa voitaisiin vielä ottaa huomioon delfiinien keskinäinen kilpailu ravinnosta ja tilasta. Merkitään K:lla maksimaalista määrää delfiinejä, jonka ympäristö voi sisältää, kutsuttakoon sitä ympäristön *kantokertoimeksi*. Kasvukerroin r korvataan ajasta ja populaation koosta riippuvalla kertoimella

$$\alpha(t) = r(1 - \frac{y(t)}{K}),$$

jolloin yhtälö saa muodon: $y' = \alpha(t)y + b(t)\,y$, missä $b(t)$ on edellisessä tehtävässä annettu, siis:

$$y' = r(1 - \frac{y}{K})y + b(t)\,y. \qquad (9)$$

Nytpä yhtälö on epälineaarinen, joten analyyttinen ratkaisumahdollisuus karkaa yhä kauemmas. Funktiotiedosto `Delfiinitfinaali.m` laskee ja piirtää sekä Eulerin menetelmällä että OCTAVE:n ennen mainitulla `ode45`-ratkaisijalla. Näytetään edellinen, jossa on piirretty näkyviin myös laskentapisteet.



Funktio: `Delfiinitfinaali.m`

Tällä kertaa kyseessä **ei ole skripti**, vaan **funktio**, joten kutsu tapahtuu komentoriviltä.

Molemmat, Eulerilla ja `ode45`:llä lasketut saat komennolla:

```
>> [ye,yo]=Delfiinitfinaali(50)
ye =
   80.1161  % ye laskettiin Eulerilla
yo =
   80.1395  % yo ode45:llä
```

Huomionarvoista on, että tulokset poikkeavat vasta neljännessä numerossa, vaikka Eulerin menetelmällä otetaan vain 50 askelta.

Tässä tapauksessa ratkaisukäyrä on hiukan aaltoileva suora, jolloin on helppo uskoa, että Eulerin menetelmäkin pääsee hyvään tarkkuuteen pienellä askelmäärällä.

Delfiinien kannalta ei kylläkään hyvältä näytä, ainoa toivo on, että valitut parametrit eivät vastaisi todellisuutta ainakaan koko aikavälillä, ja/tai muutos parempaan tapahtuisi viimeistään elokuussa 2025.

Viitteet

[QG] Alfio Quarteroni – Paola Gervasio: A Primer on Mathematical Modelling, Springer 2020. https://www.springer.com/gp/book/9783030445409

[HW] J.H. Hubbard – B.H. West: Differential Equations, A Dynamical Systems Approach, Part 1, Springer 1991.

[OctOL] https://octave-online.net/

[OctOh] Eaton, Bateman, Hauberg,Wehbring 2019 GNU Octave manual: a high-level interactive language for numerical computations. https://octave.org/doc/v5.1.0/

[AL] https://math.aalto.fi/∼apiola/matlab/opas/lyhyt/

[CF] Juhani Pitkäranta: Calculus Fennicus, kirjana loppuunmyyty, e-kirja ladattavissa: https://github.com/avoimet-oppimateriaalit-ry/calculus-fennicus/releases

[OLehto] Olli Lehto: Tieteen aatelia : Lorenz Lindelöf ja Ernst Lindelöf, Otava 2008

[Skr] Kirjoitukseen liittyvät skriptit ja funktiotiedostot: https://matematiikkalehtisolmu.fi/2021/1/skriptit/