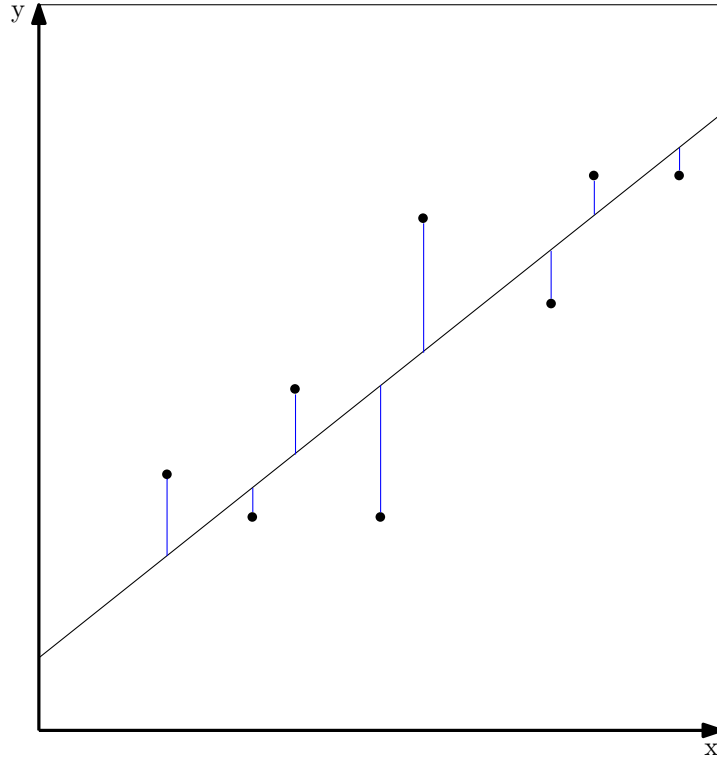


Taivaanmekaniikka
Lineaarinen pienimmän neliösumman sovitus

Anni Järvenpää

25. lokakuuta 2015



Kuva 1: Pistejoukko, johon sovitettu suora mustalla ja pisteiden vertikaaliset etäisyydet suorasta sinisellä.

1 Lineaarinen pienimmän neliösumman menetelmä

?? Lineaarisen pienimmän neliösumman menetelmän tavoitteena on sovittaa n muotoa (x_i, y_i) olevasta pisteestä koostuvaan havaintoaineistoon suora, joka edustaa pisteitä mahdollisimman hyvin. Tyypillisesti sovituksen hyvyttä mitataan pisteiden vertikaalisena etäisyytenä $|e|$ sovitetusta suorasta (merkitty sinisellä kuvassa 1). Pienimmän neliösumman menetelmässä näiden vertikaalisten poikkeamien neliöiden summa pyritään minimoimaan, siis etsimään funktion $f(x) = \beta_1 + \beta_2 x + \beta_3 x^2 + \dots + \beta_{m+1} x^m$ kertoimet $\beta_1 \dots \beta_{m+1}$ siten, että virhe S on mahdollisimman pieni, kun S on määritelty yhtälön 1 mukaisesti. [3]

$$S = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - f(x_i))^2 \quad (1)$$

Virheen minimiarvot löytyvät derivaatan nollakohdista: [1]

$$\begin{aligned}\frac{\partial R^2}{\partial \beta_1} &= -2 \sum_{i=1}^n [y - (\beta_1 + \beta_2 x + \dots + \beta_{m+1} x^m)] = 0 \\ \frac{\partial R^2}{\partial \beta_2} &= -2 \sum_{i=1}^n [y - (\beta_1 + \beta_2 x + \dots + \beta_{m+1} x^m)] = 0 \\ &\vdots \\ \frac{\partial R^2}{\partial \beta_{m+1}} &= -2 \sum_{i=1}^n [y - (\beta_1 + \beta_2 x + \dots + \beta_{m+1} x^m)] = 0\end{aligned}$$

Näistä saadaan edelleen

$$\begin{aligned}\beta_1 n + \beta_2 \sum_{i=1}^n x_i + \dots + \beta_{m+1} \sum_{i=1}^n x_i^m &= \sum_{i=1}^n y_i \\ \beta_1 \sum_{i=1}^n x_i + \beta_2 \sum_{i=1}^n x_i^2 + \dots + \beta_{m+1} \sum_{i=1}^n x_i^{m+1} &= \sum_{i=1}^n x_i y_i \\ \beta_1 \sum_{i=1}^n x_i^m + \beta_2 \sum_{i=1}^n x_i^{m+1} + \dots + \beta_{m+1} \sum_{i=1}^n x_i^{2m} &= \sum_{i=1}^n x_i^m y_i\end{aligned}$$

eli matriisimuodossa

$$\begin{bmatrix} n & \sum_{i=1}^n x_i & \dots & \sum_{i=1}^n x_i^m \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 & \dots & \sum_{i=1}^n x_i^{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^n x_i^m & \sum_{i=1}^n x_i^{m+1} & \dots & \sum_{i=1}^n x_i^{2m} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \\ \beta_{m+1} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n y_i \\ \sum_{i=1}^n y_i x_i \\ \vdots \\ \sum_{i=1}^n y_i x_i^m \end{bmatrix}. \quad (2)$$

Voidaan myös huomata, että yhtälöön 2 voidaan päästä hyödyntämällä Vandermonden matriisia [2]

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^n \end{bmatrix} \quad (3)$$

ja kirjoittamalla tämän avulla

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ 1 & x_3 & x_3^2 & \dots & x_3^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{m+1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}. \quad (4)$$

Nyt voidaan kertoa edellinen yhtälö puolittain vasemmalta Vandermonden matriisin transpoosilla \mathbf{X}^T , jolloin yhtälö saadaan seuraavaan muotoon:

$$\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ x_1 & x_2 & x_3 & \dots & x_m \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_m^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^n & x_2^n & x_3^n & \dots & x_m^n \end{bmatrix} \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ 1 & x_3 & x_3^2 & \dots & x_3^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{m+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1^2 & x_2^2 & \dots & x_m^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^n & x_2^n & \dots & x_m^n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

Suoritetaan yhtälön matriisien kertolaskut, jolloin saadaan

$$\begin{bmatrix} n & \sum_{i=1}^n x_i & \dots & \sum_{i=1}^n x_i^m \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 & \dots & \sum_{i=1}^n x_i^{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^n x_i^m & \sum_{i=1}^n x_i^{m+1} & \dots & \sum_{i=1}^n x_i^{2m} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{m+1} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n y_1 \\ \sum_{i=1}^n y_2 \\ \vdots \\ \sum_{i=1}^n y_m \end{bmatrix}. \quad (5)$$

Nyt voidaan huomata yhtälöiden 5 ja 2 olevan identtiset. Koska Vandermonden matriisi on kääntyvä¹, on kaikille yhtälöstä 4 yhtälöön 5 pääsemiseksi tehdyille operaatioille käänteisoperaatio. Täten yhtälöt 4 ja 2 ovat yhtäpitävät, eli pienimmän neliösumman antavat kertoimet β_i voidaan ratkaista yhtälöstä 4.

Otetaan kerroinvektorin ratkaisemisen helpottamiseksi käyttöön seuraavat merkinnät:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad \text{ja} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix},$$

¹Mikä voidaan helposti osoittaa, ks. esim https://proofwiki.org/wiki/Inverse_of_Vandermonde's_Matrix

jolloin yhtälö 2 voidaan kirjoittaa lyhyesti

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} \quad (6)$$

ja tästä ratkaista $\boldsymbol{\beta}$ seuraavasti:

$$\begin{aligned} \mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} & \|\mathbf{X}^T. \\ \mathbf{X}^T\mathbf{Y} &= \mathbf{X}^T\mathbf{X}\boldsymbol{\beta} & \|(\mathbf{X}^T\mathbf{X}\boldsymbol{\beta})^{-1} \\ \boldsymbol{\beta} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \end{aligned} \quad (7)$$

Samankaltaisella päättelyketjulla voidaan ratkaista yleinen lauseke myös tilanteelle, jossa kaikkien pisteiden mittaustarkkuus ei ole sama, jolloin pisteille halutaan käyttää erilaisia painokertoimia sovitusta tehtäessä. Tähän voidaan käyttää varianssimatriisia \mathbf{V} , jonka avulla voidaan määritellä painokerroinmatriisi $\mathbf{W} = \mathbf{V}^{-1}\sigma^2$. Tässä σ on tutkittavan aineiston varianssi ja \mathbf{V} määritelty seuraavasti: [3]

$$\mathbf{V} = \begin{bmatrix} \sigma_1^2 & \sigma_{1,2} & \dots & \sigma_{1,n} \\ \sigma_{2,1} & \sigma_2^2 & \dots & \sigma_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n,1} & \sigma_{n,2} & \dots & \sigma_n^2 \end{bmatrix}$$

Näitä merkintöjä käyttäen kerroinvektori $\boldsymbol{\beta}$ voidaan ratkaista yhtälöstä 8 [3].

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{Y} \quad (8)$$

2 Pienimmän neliösumman ratkaisu kaksiulotteiselle havaintoaineistolle

Kun pienimmän neliösumman sovitusta sovelletaan asteroidin radanmäärittelyyn, on käytössä kaksiulotteinen havaintoaineisto, joka koostuu asteroidin paikoista (x_i, y_i) ajanhetkellä t_i . Asteroidin paikkavektori $\mathbf{r}(t)$ mielivaltaisella ajanhetkellä t voidaan ratkaista asteroidin rataelementtien avulla. Rataelementeiksi voidaan valita asteroidin paikka $\mathbf{R}_0 = (X_0, Y_0)$ ja nopeus $\mathbf{V}_0 = (\dot{X}_0, \dot{Y}_0)$ ajanhetkellä $t = 0$, jolloin asteroidin paikka saadaan seuraavasti:

$$\mathbf{r}(t) = \mathbf{R}_0 + \mathbf{V}_0 t. \quad (9)$$

Havaintoaineiston ollessa kaksiulotteinen, täytyy \mathbf{Y} -vektorin tallentaa sekä x - että y -

koordinaatit, jolloin

$$\mathbf{Y} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_n \\ y_n \end{bmatrix}.$$

Koska tahdotaan ratkaista aiemmin määritellyt rataelementit, on myös ratkaistava kerroinvektori erilainen:

$$\boldsymbol{\beta} = \begin{bmatrix} X_0 \\ Y_0 \\ \dot{X}_0 \\ \dot{Y}_0 \end{bmatrix}.$$

Myös havainnot sisältävän matriisin \mathbf{X} täytyy mukautua kaksiulotteiseen havaintoaineistoon, jotta kertolasku \mathbf{Y} -vektorin kanssa on mahdollinen ja fysikaalisesti mielekäs. Tähän sopii alla esitetty matriisi, jolla $(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$ tuottaa kooltaan $4 \times 2n$ matriisin, joka voidaan edelleen kertoa edellä määritellyllä vektorilla \mathbf{Y} , jolloin saadaan kooltaan matriisia $\boldsymbol{\beta}$ vastaava matriisi. Ajanhetki t_i on pisteen (x_i, y_i) havaintoaika.

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & t_1 & 0 \\ 0 & 1 & 0 & t_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & t_n & 0 \\ 0 & 1 & 0 & t_n \end{bmatrix}$$

Näillä uudelleenmääritellyillä on mahdollista käyttää suoraan aiemmin ratkaistuja yhtälöitä 7 (kaikkien havaintojen painokerroin sama) tai 8 (painokertoimet). Mikäli käytetään painokertoimia, on myös painokerroinmatriisin koko luonnollisesti $2n \times 2n$. Ratkaisuna saatavasta vektorista $\boldsymbol{\beta}$ saadaan suoraan rataelementit \mathbf{R}_0 ja \mathbf{V}_0 .

3 Ennusteen virhearvio lineaarisessa mallissa

Yhtälön 8 mukaisesti ratkaistujen parametrien $\boldsymbol{\beta}$ virhe voidaan laskea kun tiedetään, että lineaarikombinaatiossa $\mathbf{P}^T \mathbf{Y}$, missä \mathbf{P} on vakiovektori, saadaan varianssimatriisi seuraavasti [4]:

$$\text{var}(\mathbf{P}^T \mathbf{Y}) = \mathbf{P}^T \text{var}(\mathbf{Y}) \mathbf{P}.$$

Tällöin saadaan

$$\text{var}(\boldsymbol{\beta}) = (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} \text{var}(\boldsymbol{\beta}) \mathbf{W} (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1}. \quad (10)$$

Hyödyntämällä aiemmin määriteltyä $\mathbf{W} = \mathbf{V}^{-1} \sigma^2$ ja tietoa $\text{var}(\boldsymbol{\beta}) = \mathbf{V}$ eli $\text{var}(\boldsymbol{\beta}) = \sigma^2 \mathbf{W}^{-1}$ saadaan yhtälö 10 muotoon

$$\text{var}(\boldsymbol{\beta}) = \sigma^2 (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \quad (11)$$

Kun parametrien β_i varianssit σ_i tunnetaan, saadaan niistä helposti keskihajonnat s (yhtälö 12) tai keskiarvon keskivirheet $s_{\bar{x}}$ (yhtälö 13) [5].

$$\sigma = s^2 \quad (12)$$

$$s_{\bar{x}} = \frac{s}{\sqrt{n}} \quad (13)$$

Kun yksittäisten parametrien virheet tiedetään, voidaan niiden perusteella laskettavan funktion f virhe määrittää käyttäen virheen kasautumislakia [6]

$$\delta f = \sqrt{\sum_{i=1}^n \left(\frac{\partial f}{\partial \beta_i} \delta \beta_i \right)^2}, \quad (14)$$

missä virheenä käytetään tyypillisesti keskiarvon keskivirhettä.

Virheen kasautumislakia käyttäen saadaan luvun 2 yhtälössä 9 esitetylle asteroidin paikalle ajan funktiona virhe seuraavasti:

$$\begin{aligned} \delta r(t) &= \sqrt{\left(\frac{\partial r}{\partial X_0} \delta X_0 \right)^2 + \left(\frac{\partial r}{\partial Y_0} \delta Y_0 \right)^2 + \left(\frac{\partial r}{\partial \dot{X}_0} \delta \dot{X}_0 \right)^2 + \left(\frac{\partial r}{\partial \dot{Y}_0} \delta \dot{Y}_0 \right)^2} \\ &= \sqrt{\delta X_0^2 + \delta Y_0^2 + \left(\delta \dot{X}_0 t \right)^2 + \left(\delta \dot{Y}_0 t \right)^2} \end{aligned} \quad (15)$$

ja vastaavasti koska $v(t) = \frac{dr}{dt} = (\dot{X}_0, \dot{Y}_0)$, saadaan

$$\begin{aligned} \delta v(t) &= \sqrt{\left(\frac{\partial v}{\partial \dot{X}_0} \delta \dot{X}_0 \right)^2 + \left(\frac{\partial v}{\partial \dot{Y}_0} \delta \dot{Y}_0 \right)^2} \\ &= \sqrt{\delta \dot{X}_0^2 + \delta \dot{Y}_0^2} \end{aligned} \quad (16)$$

4 Python-implementaatio

Toteutin lisäksi Pythonilla ohjelman, joka sovittaa aiemmin esitettyä menetelmää käyttäen n . asteen polynomin annettuihin datapisteisiin. Liitteessä A esitelty ohjelma ottaa komentoriviargumentteina sovittettavan käyrän asteen sekä tiedostot, joista se lukee sovituksessa käytettävät matriisit \mathbf{X} , \mathbf{Y} ja \mathbf{W} . Matriisiksi \mathbf{X} voi antaa joko yksisarakeisen vektorin, jolloin sen oletetaan olevan yhtälön 3 toinen sarake, tai kokonaisen matriisin, jolloin sitä käytetään sellaisenaan. Tämä mahdollistaa ohjelman käytön myös kaksiulotteiselle havaintoaineistolle. Matriisia \mathbf{Y} käytetään aina sellaisenaan.

Ohjelman testaamiseksi generoin funktiolla $y = 1.5x^3 - 5x^2 + 3x + 1.5$ sata datapistettä väliltä $[-1, 4]$, yhden setin keskihajonnalla 3 ja toisen keskihajonnalla 0.5. Sen jälkeen sovitin kumpaankin datasettiin kolmannen asteen käyrän käyttäen painokerroinmatriisina yksikkömatriisia, sillä kaikkien pisteiden ”mittaustarkkuus” on sama. Pisteet ja niihin sovitetut käyrät ovat nähtävillä kuvissa 2 ja 3. Keskihajonnalla 3 saatiin kerroinmatriisi

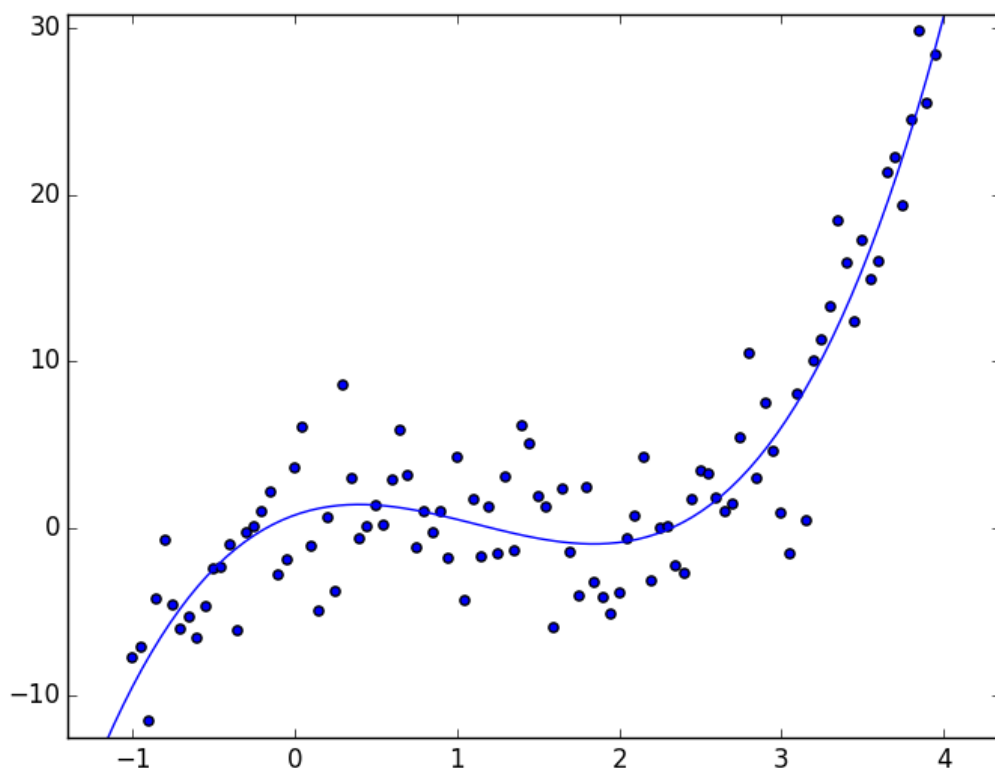
$$\beta_3 = \begin{bmatrix} 0.78875931 \\ 3.45233752 \\ -5.28074743 \\ 1.5722111 \end{bmatrix}$$

ja keskihajonnalla 0,5

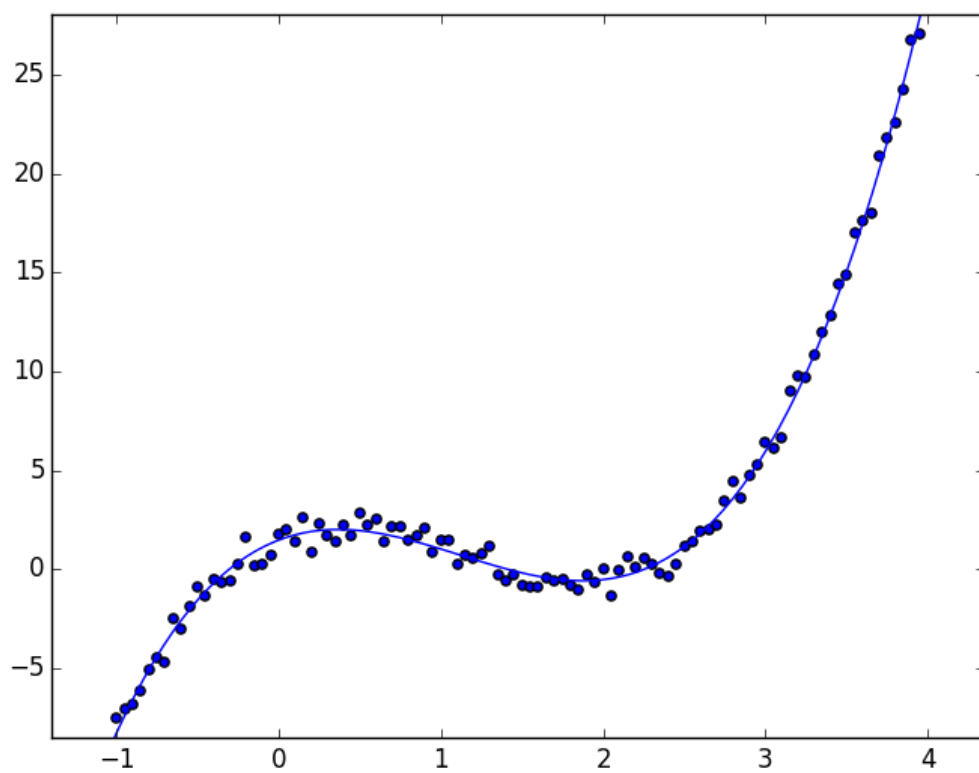
$$\beta_{0.5} = \begin{bmatrix} 1,48549344 \\ 3,13099507 \\ -5,10374017 \\ 1,51945966 \end{bmatrix}.$$

Näistä jälkimmäinen on selvästi lähempänä dataa generoitaessa käytettyä matriisia

$$\beta_{\text{tod}} = \begin{bmatrix} 0.5 \\ 3 \\ -5 \\ 1.5 \end{bmatrix}.$$



Kuva 2: Keskihajonnalla 3 generoidut pisteet ja niihin sovitettu käyrä



Kuva 3: Keskihajonnalla 0.5 generoidut pisteet ja niihin sovitettu käyrä

Viitteet

- [1] Least squares fitting-polynomial. <http://mathworld.wolfram.com/LeastSquaresFittingPolynomial.html>. Luettu: 15.10.2015.
- [2] Vandermonde matrix. <http://mathworld.wolfram.com/VandermondeMatrix.html>. Luettu: 16.10.2015.
- [3] Peter J. Bickel. *Mathematical statistics: basic ideas and selected topics*. Holden-Day, Oakland, CA, 1977.
- [4] B. R. Martin. *Statistics for Physicists*. Physicists Academic Press Inc., 1971.
- [5] Raimo Seppänen. *MAOL-taulukot : matematiikka, fysiikka, kemia*. Otava, Helsingissä, 2005. Lisäpainokset: 2.-3.: p. 2006. - 4.-5. p. 2007. - 2.-9. p. 2011.
- [6] John R. Taylor. *An introduction to error analysis : the study of uncertainties in physical measurements*. University Science Books, Sausalito, CA, 1997. Sisältää hakemiston.

A Käytetty ohjelma

A.1 pns.py

```
1  # -*- coding: utf-8 -*-
2  import numpy as np
3  import matplotlib.pyplot as plt
4  import sys
5  from utils import *
6
7  if len(sys.argv) != 5 :
8      print "Anna komentoriviargumentteina sovitettavan käyrän
9          aste sekä tiedostot, joissa käytettävät matriisit X, Y
10         ja W ovat"
11      sys.exit(1)
12
13  aste = int(sys.argv[1])
14  xHavainto = lueMatriisi(sys.argv[2])
15  Y = lueMatriisi(sys.argv[3])
16  W = lueMatriisi(sys.argv[4])
17
18  if len(np.transpose(xHavainto)) == 1 : # jos annetussa
19      X-matriisissa on vain yksi sarake, oletetaan n. asteen käyrä
20      y=f(x) ja luodaan X-matriisi sen mukaan
21      X = np.ones((len(xHavainto), aste+1))
22      for i in range(0, len(xHavainto)):
23          for j in range(1, aste+1):
24              X[i][j] = xHavainto[i]**j
25  else : # muuten käytetään annettua sellaisenaan
26      X = xHavainto
27
28  if len(X) != len(Y) or len(Y) != len(W):
29      print "Annettujen matriisien dimensiot eivät täsmää"
30      sys.exit(1)
31
32  beta = np.dot( np.dot( np.dot( np.linalg.inv( np.dot( np.dot(
33      np.transpose(X), W), X)), np.transpose(X)), W), Y)
34  print "kerroinmatriisi beta:"
35  print beta
```

A.2 utils.py

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 def lueMatriisi(tiedosto):
5     f = open ( tiedosto , 'r')
6     matriisi = [ map(float,line.split(',')) for line in f ]
7     return np.array(matriisi)
8
9 def plottaaFunktio(xmin, xmax, kertoimet):
10     pisteet = 100
11     x = np.arange(xmin, xmax, (xmax-xmin)/pisteet)
12     y = np.zeros(pisteet)
13
14     for i in range(0, pisteet):
15         for k in range(0, len(kertoimet)):
16             y[i] += x[i]**k*kertoimet[k]
17     plt.plot(x, y)
```