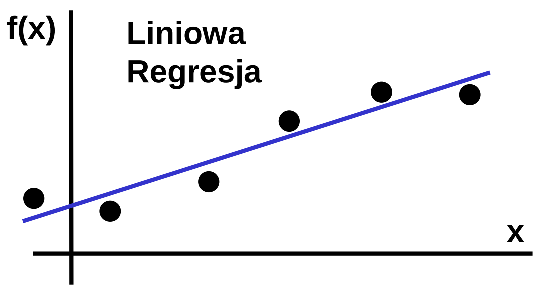
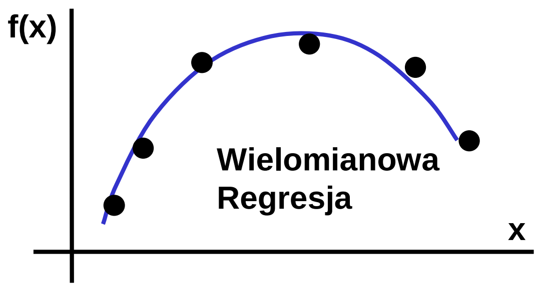
Aproksymacja – najczęściej dopasowywanie funkcji ciągłej do zbioru punktów pomiarowych

- **Regresja (pojęcie stosowane w statystyce)/aproksymacja:** Głównie używana z danymi eksperymentalnymi, które mogą mieć znaczną ilość błędów (szum). Nie szukamy funkcji przechodzącej przez wszystkie dyskretne punkty jak w przypadku interpolacji.

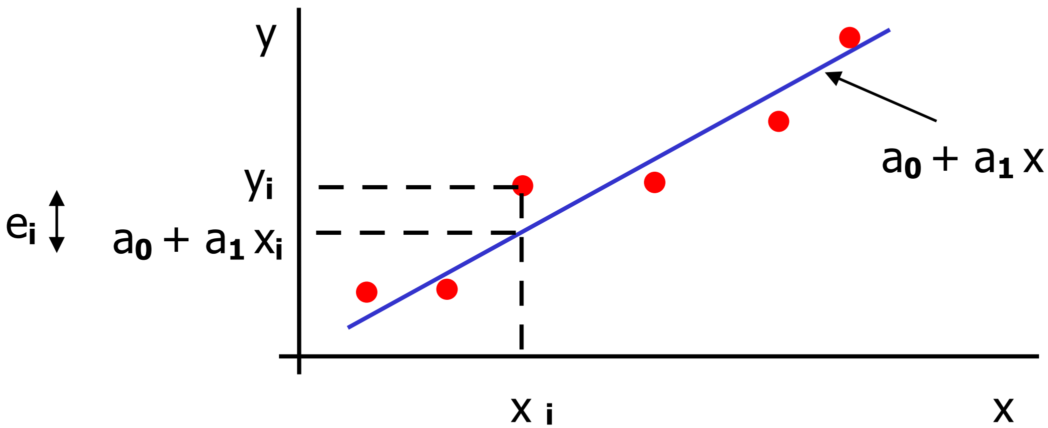




Metoda najmniejszych kwadratów – przypadek liniowy

- Dopasowanie funkcji liniowej (prostej) do zbioru punktów.

$$(x_1, y_1), \quad (x_2, y_2), \quad (x_3, y_3), \quad . . . , \quad (x_n, y_n)$$



a_0 : przecięcie osi y (nieznane)

a_1 : nachylenie (nieznane)

$e_i = y_i - a_0 - a_1 x_i$

Błąd (odchylenie) dla i[tego] punktu

- Minimalizacja błędu (odchylenia) aby otrzymać najlepsze dopasowanie (znalezienie a_0 i a_1). Możliwości:
- Minimalizacja maksymalnego błędu (aproksymacja jednostajna).
- Minimalizacja sumy kwadratów poszczególnych błędów – strategia MNKW (aproksymacja średniokwadratowa).

Minimalizacja sumy kwadratów odchyleń

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2$$

Suma kwadratów residuów (reszt)

- Określamy nieznane a_0 i a_1 poprzez minimalizację S_r.
- Obliczamy pochodne S_r względem a_0 i a_1 i przyrównujemy do zera (warunek konieczny ekstremum).

$$\frac{\partial S_r}{\partial a_0} = -2\sum_{i=1}^{n}(y_i - a_0 - a_1 x_i) = 0 \quad \rightarrow \quad n\,a_0 + \left(\sum x_i\right)a_1 = \sum y_i$$

$$\frac{\partial S_r}{\partial a_1} = -2\sum_{i=1}^{n}[(y_i - a_0 - a_1 x_i)x_i] \quad \rightarrow \quad \left(\sum x_i\right)a_0 + \left(\sum x_i^2\right)a_1 = \sum x_i y_i$$

$$\begin{bmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} \sum y_i \\ \sum x_i y_i \end{Bmatrix}$$

- Znajdujemy a_0 i a_1.

Rozwiązanie (kreski nad y i x oznaczają wartości średnie)

$$\boxed{a_1 = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2} \qquad a_0 = \bar{y} - a_1 \bar{x}}$$

Przykład 1:

Użyj regresji liniowej metodą najmniejszych kwadratów, aby dopasować linię prostą do

x	1	3	5	7	10	12	13	16	18	20
y	4	5	6	5	8	7	6	9	12	11

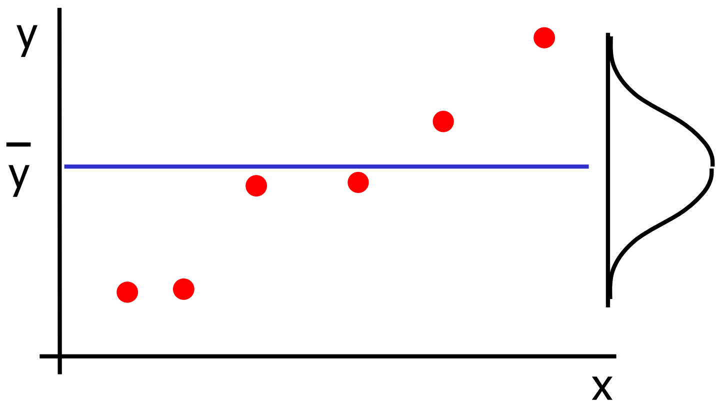
$$n = 10$$
$$\sum x_i = 105$$
$$\sum y_i = 73$$
$$\bar{x} = 10.5$$
$$\bar{y} = 7.3$$
$$\sum x_i^2 = 1477$$
$$\sum x_i y_i = 906$$

$$a_1 = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2} = \frac{10*906 - 105*73}{10*1477 - 105^2} = 0.3725$$

$$a_0 = 7.3 \text{ - } 0.3725*10.5 = 3.3888$$

Błąd regresji liniowej

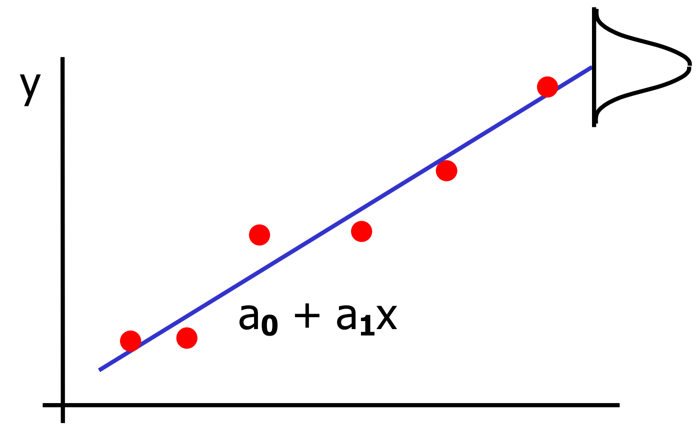
Rozrzut danych wokół średniej



$$S_t = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$s_y = \sqrt{\frac{S_t}{n-1}}$$

Rozrzut danych wokół linii regresji



$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1x_i)^2$$

$$s_{y/x} = \sqrt{\frac{S_r}{n-2}}$$

- Poprawa uzyskana za pomocą linii regresji zamiast średniej wartości daje miarę tego, jak dobre jest dopasowanie regresji.

współczynnik determinacji

$$r^2 = \frac{S_t - S_r}{S_t}$$

współczynnik korelacji

$$r = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}$$

Jak interpretować współczynnik korelacji?

- Dwa skrajne przypadki:
- $S_r = 0$ -> $r=1$ opisuje idealne dopasowanie (prosta linia przechodząca przez wszystkie punkty).
- $S_r = S_t$ -> $r=0$ całkowity brak korelacji pomiędzy x i y.

- Zwykle wartość r bliska 1 oznacza dobre dopasowanie.

Przykład 2:

Oblicz współczynnik korelacji dla poprzedniego przykładu.

$$
\begin{aligned}
&n = 10 \\
&\sum x_i = 105 \\
&\sum y_i = 73 \\
&\bar{x} = 10.5 \\
&\bar{y} = 7.3 \\
&\sum x_i^2 = 1477 \\
&\sum x_i y_i = 906
\end{aligned}
$$

$$
\begin{aligned}
&S_t = \sum_{i=1}^{n} (y_i - \bar{y})^2 = 64.1 \\
&S_r = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2 = 12.14 \\
&r^2 = \frac{S_t - S_r}{S_t} = 0.8107 \\
&r = 0.9
\end{aligned}
$$

Przykład 3:

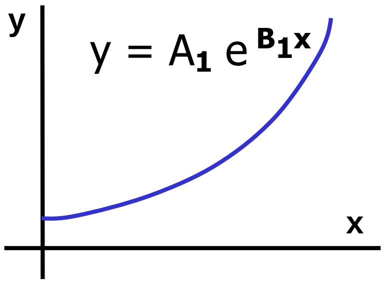
Odwróć x i y. Znajdź liniową linię regresji i oblicz r.

$x = -5.3869 + 2.1763\, y$
$S_t = 374.5$, $S_r = 70.91$ (inne niż wcześniej).
$r^2 = 0.8107$, $r = 0.9$ (takie samo jak wcześniej).

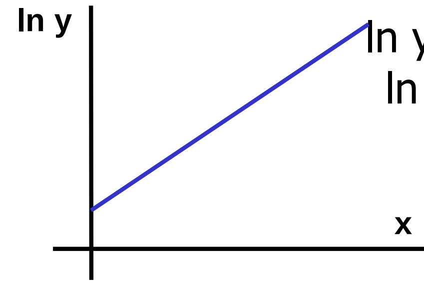
Linearyzacja zachowania nieliniowego

- Regresja liniowa jest przydatna do reprezentowania zależności liniowej.
- Jeśli relacja jest nieliniowa, można zastosować inną technikę lub dane można przekształcić, aby można było nadal stosować regresję liniową. Ta ostatnia technika jest często używana do dopasowania poniższych równań nieliniowych do zbioru danych..
- Równanie wykładnicze ($y=A_1 e^{B_1 x}$)
- Równanie potęgowe ($y=A_2 x^{B_2}$)
- Równanie szybkości nasycenia i wzrostu ($y=A_3 x / (B_3+x)$)

(1) Równanie wykładnicze ($y = A_1 e^{B_1 x}$)



Linearyzacja $\Longrightarrow$



$\ln y = \ln A_1 + B_1 x$

$\ln y = a_0 + a_1 x$

Przykład 4: Dopasuj model wykładniczy do następującego zestawu danych.

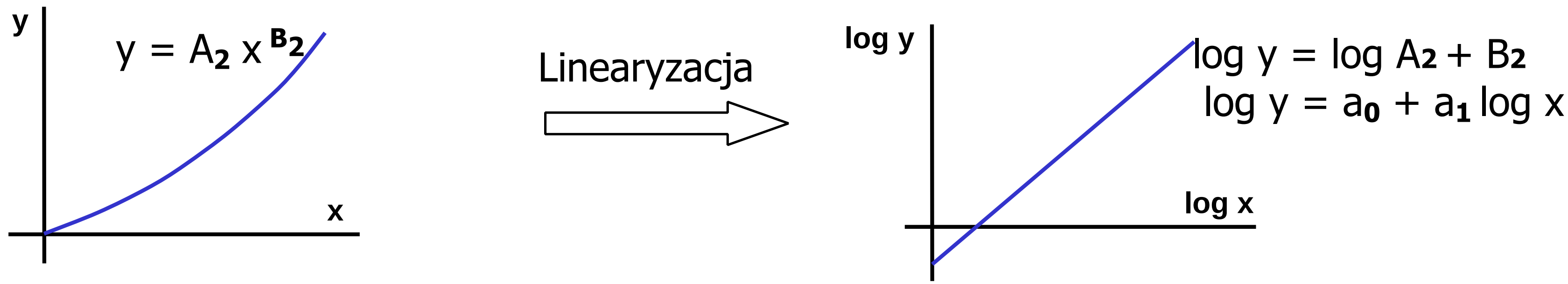
x	0.4	0.8	1.2	1.6	2.0	2.3
y	750	1000	1400	2000	2700	3750

- Utwórz poniższą tabelę.

x	0.4	0.8	1.2	1.6	2.0	2.3
ln y	6.62	6.91	7.24	7.60	7.90	8.23

- Dopasuj linię prostą do tego nowego zestawu danych. Zachowaj ostrożność z notacją. Możesz zdefiniować z = ln y
- Oblicz $a_0 = 6.25$ i $a_1 = 0.841$. Równanie prostej $\ln y = 6.25 + 0.841\, x$
- Przejdź z powrotem do pierwotnego równania. $A_1 = e^{a_0} = 518$, $B_1 = a_1 = 0.841$.
- Dlatego równanie wykładnicze jest $y = 518\, e^{0.841x}$. Sprawdź to rozwiązanie z kilkoma punktami danych. Na przykład $y(1.2) = 518\, e^{0.841\,(1.2)} = 1421$ or $y(2.3) = 518\, e^{0.841\,(2.3)} = 3584$. OK.

(2) Równanie potęgowe ($y = A_2 x^{B_2}$)



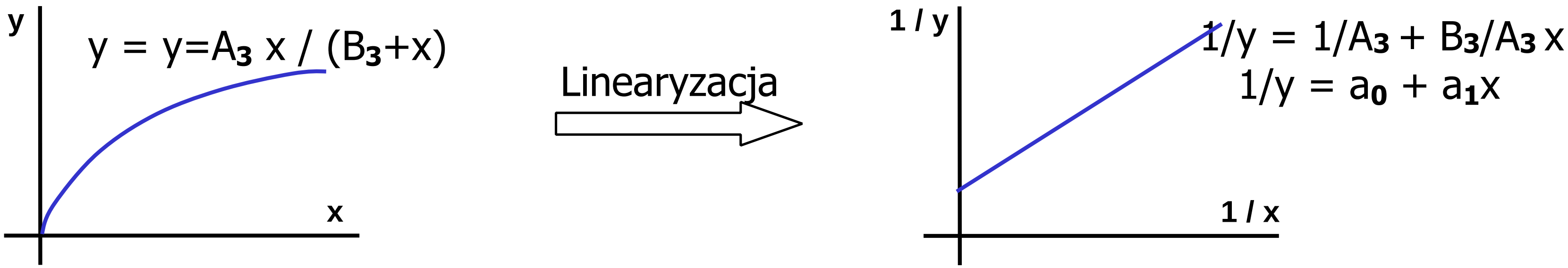
Przykład 5: Dopasuj model potęgowy do następującego zestawu danych.

x	2.5	3.5	5	6	7.5	10	12.5	15	17.5	20
y	7	5.5	3.9	3.6	3.1	2.8	2.6	2.4	2.3	2.3
log x	0.398	0.544	0.699	0.778	0.875	1.000	1.097	1.176	1.243	1.301
log y	0.845	0.740	0.591	0.556	0.491	0.447	0.415	0.380	0.362	0.362

- Dopasuj linię prostą do tego nowego zestawu danych. Zachowaj ostrożność z notacją.
- Oblicz $a_0 = 1.002$ i $a_1 = -0.53$. Równanie prostej $\log y = 1.002 - 0.53 \log x$

- Przejdź z powrotem do pierwotnego równania. $A_2 = 10^{a_0} = 10.05$, $B_2 = a_1 = -0.53$.
- Dlatego równanie potęgowe $y = 10.05\ x^{-0.53}$. Sprawdź to rozwiązanie z kilkoma punktami danych. Na przykład $y(5) = 10.05 * 5^{-0.53} = 4.28$ lub $y(15) = 10.05 * 15^{-0.53} = 2.39$. OK.

(3) Równanie szybkości nasycenia i wzrostu ($y = A_3 x / (B_3+x)$)



Przykład 6: Dopasuj model szybkości nasycenia i wzrostu do następującego zestawu danych.

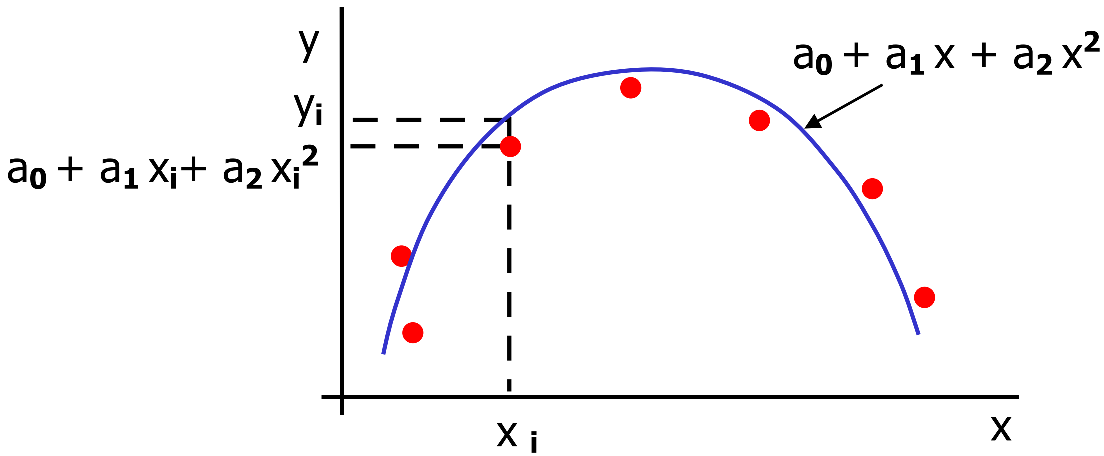
x	0.75	2	2.5	4	6	8	8.5
y	0.8	1.3	1.2	1.6	1.7	1.8	1.7
1 / x	1.333	0.5	0.4	0.25	0.1667	0.125	0.118
1 / y	1.25	0.769	0.833	0.625	0.588	0.556	0.588

- Dopasuj linię prostą do tego nowego zestawu danych. Zachowaj ostrożność z notacją.
- Oblicz $a_0 = 0.512$ i $a_1 = 0.562$. Równanie prostej $1/y = 0.512 + 0.562\,(1/x)$

- Przejdź z powrotem do pierwotnego równania. $A_3 = 1/a_0 = 1.953$, $B_3 = a_1 A_3 = 1.097$.
- Dlatego równanie pierwotne $1/y = 1.953\, x/(1.097+x)$. Sprawdź to rozwiązanie z kilkoma punktami danych. Na przykład $y(2) = 1.953*2/(1.097+2) = 1.26$ OK.

Regresja wielomianowa (rozszerzenie metody najmniejszych kwadratów liniowych)

Używana do znalezienia wielomianu najlepszego dopasowania dla zachowania nieliniowego.



Regresja wielomianowa drugiego stopnia

$e_i = y_i - a_0 - a_1 x_i - a_2 x_i^2$

Błąd dla i-tego punktu

Suma kwadratów residuów

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i - a_2 x_i^2)^2$$

- Minimalizujemy tą sumę i otrzymujemy

$$\frac{\partial S_r}{\partial a_0} = 0, \quad \frac{\partial S_r}{\partial a_1} = 0, \quad \frac{\partial S_r}{\partial a_2} = 0$$

- Następnie znajdujemy a_0, a_1 i a_2.

Przykład 7: Znajdź parabolę, korzystając z metody najmniejszych kwadratów, która najlepiej pasuje do następującego zestawu danych.

x	0	1	2	3	4	5
y	2.1	7.7	13.6	27.2	40.9	61.1

- Równania zgodnie z metodą najmniejszych kwadratów

$$\begin{bmatrix} n & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \sum y_i \\ \sum x_i y_i \\ \sum x_i^2 y_i \end{Bmatrix}$$

$n = 6$

$\sum x_i = 15 \qquad \sum y_i = 152.6$

$\sum x_i^2 = 55 \qquad \sum x_i y_i = 585.6$

$\sum x_i^3 = 225 \qquad \sum x_i^2 y_i = 2488.6$

$\sum x_i^4 = 979$

$\longrightarrow$

$a_0 = 2.479, \quad a_1 = 2.359, \quad a_2 = 1.861$

$y = 2.479 + 2.359\,x + 1.861\,x^2$

$$r^2 = \frac{S_t - S_r}{S_t} = \frac{2573.4 - 3.75}{2573.4} = 0.999$$

$r = 0.999$

Wielokrotna regresja liniowa

$y = y(x_1, x_2)$

- Indywidualne błędy $e_i = y_i - a_0 - a_1 x_{1i} - a_2 x_{2i}$

Suma kwadratów residuów

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_{1i} - a_2 x_{2i})^2$$

- Minimalizujemy tę sumę $\frac{\partial S_r}{\partial a_0} = 0, \quad \frac{\partial S_r}{\partial a_1} = 0, \quad \frac{\partial S_r}{\partial a_2} = 0$

$$\begin{bmatrix} n & \sum x_{1i} & \sum x_{2i} \\ \sum x_{1i} & \sum x_{1i}^2 & \sum x_{1i} x_{2i} \\ \sum x_{2i} & \sum x_{1i} x_{2i} & \sum x_{2i}^2 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \sum y_i \\ \sum x_{1i} y_i \\ \sum x_{2i} y_i \end{Bmatrix}$$

- Znajdujemy a_0, a_1 i a_2.

Przykład 8:

Użyj wielokrotnej regresji liniowej w celu dopasowania punktów

x	0	1	1	2	2	3	3	4	4
y	0	1	2	1	2	1	2	1	2
z	15	18	12.8	25.7	20.6	35	29.8	45.5	40.3

$$n = 9$$

$$\sum x_i = 20 \qquad \sum x_i y_i = 30$$

$$\sum x_i^2 = 60 \qquad \sum z_i = 242.7$$

$$\sum y_i = 12 \qquad \sum x_i z_i = 661$$

$$\sum y_i^2 = 20 \qquad \sum y_i z_i = 331.2$$

$$\longrightarrow \begin{bmatrix} 9 & 20 & 12 \\ 20 & 60 & 30 \\ 12 & 30 & 20 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 242.7 \\ 661 \\ 331.2 \end{Bmatrix}$$

$$a_0 = 14.40, \quad a_1 = 9.03, \quad a_2 = -5.62$$

$$z = 14.4 + 9.03\,x - 5.62\,y$$

Python – różne rodzaje aproksymacji średniokwadratowej

Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\u46\.spyder-py3

Editor - C:\Users\u46\.spyder-py3\fit.py

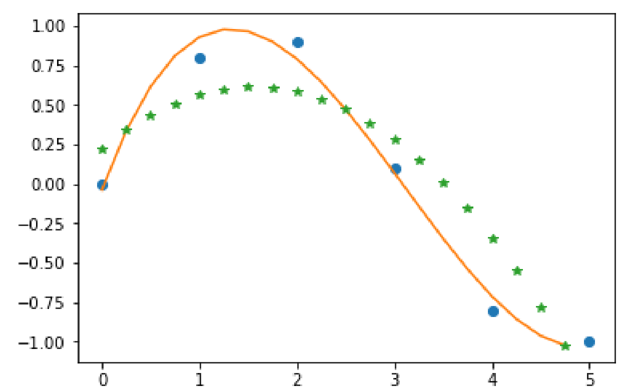
fit.py interpol.py

```python
import numpy as np
import matplotlib.pyplot as plt


x = np.array([0.0, 1.0, 2.0, 3.0,  4.0,  5.0])
y = np.array([0.0, 0.8, 0.9, 0.1, -0.8, -1.0])
z3 = np.polyfit(x, y, 3)
z2 = np.polyfit(x, y, 2)

w3 = np.poly1d(z3)
w2 = np.poly1d(z2)
xnew = np.arange(0, 5, 0.25)
ynew3=w3(xnew)
ynew2=w2(xnew)


plt.plot(x, y, 'o', xnew, ynew3,'-', xnew, ynew2,'*')

```

IPython console

Console 1/A

```
In [18]: runfile('C:/Users/u46/.spyder-py3/fit.py',
wdir='C:/Users/u46/.spyder-py3')
```




```
In [19]:
```