# Estimasi Parameter Regresi Polinomial dengan Metode Newton Raphson dan Matriks Hessian

Shafida Afifah Firdausy
Universitas Negeri Malang, Jl. Ambarawa No.5, Malang 65145, Indonesia

**ABSTRAK**

Artikel ini akan membahas estimasi parameter regresi polinomial derajat 2 dengan metode Newton Rapshon. Tujuan dari artikel ini untuk menemukan akar-akar dari persamaan regresi polinomial berderajat 2. Lalu akan dilihat seberapa banyak error dengan metode Newton Raphson. Lalu di akhir akan dibandingkan dengan metode regresi linier dan regresi polinomial tanpa Newton Raphson.

***Keywords:***

Newton Raphson
Estimasi
Regresi Polinomial
Metode Numerik
Matriks Hessian
Regresi Non-Linier

***Author Correspondence:***

Shafida Afifah Firdausy,
Departemen Matematika
Universitas Negeri Malang
Jl. Ambarawa No.5
E-mail: shafida.afifah.2103126@students.um.ac.id

## PENDAHULUAN

Estimasi parameter regresi polinomial atau bisa disebut juga cara memperoleh akar-akar dari polinomial aalah sebuah permasalahan yang non-linier. Permasalahan non-linier sulit diselesaikan dengan cara-cara biasa. Di sinilah implementasi metode numerik akan digunakan(Nocedal & Wright, 2000). Salah satu cara untuk mengestimasi akar-akar polinomial adalah dengan metode Newton Raphson. Metode Newton Raphson sendiri adalah pengembangan dari deret taylor.

Tujuan dari artikel ini adalah untuk menghitung parameter regresi polinomial dengan menggunakan metode Newton Raphson. Metode Newton Raphson sering digunakan untuk menyelesaikan masalah *rootfinding*. *Rootfinding* ini adalah cara menemukan akar persamaan, sesuai dengan namanya yang terdiri dari kata *root* yang artinya akar dan *finding* yang artinya temuan. Dalam artikel ini akan digunakan data harga rumah. Lalu data harga rumah akan dimodelkan dengan regresi polinomial. Setelah itu akan dilihat nilai galatnya. Lalu akan ditentukan apakah metode Newton Raphson efektif dalam mengestimasi nilai nilai akar regresi polinomial. Lalu, di akhir penelitian ini akan dibandingkan metode regresi linier, regresi polinomial dan regresi polinomial metode Newton Raphson.

Penelitian tentang penggunaan metode Newton Raphson telah dilakukan oleh Gireesh Kumar dkk., pada tahun 2023. Penelitian ini berjudul "*Design of an Optimized Asymmetric Multilevel Inverter with Reduced Components Using Newton-Raphson Method and Particle Swarm Optimization*". Penelitian juga telah dilakukan oleh (Otkun, 2021) dengan judul "*Newton–Raphson based scalar speed control and optimization of IM*". Penelitian ini memakai metode Newton Raphson dengan scalar speed control(SSC) untuk mengoptimalkan induksi motor(IM).

Sedangkan, penelitian tentang regresi non-linier telah dilakukan oleh Ali dkk., pada tahun 2020 yang berjudul tentang "*Estimation of Non-Linear Regression Parameters by Newton Raphson's Method of Nonlinear Equations*". Pada penelitian dijelaskan bagaiman penulis dapat mengimplementasi metode Newton Raphson dengan

## PEMBAHASAN

### Materi Prasyarat

1. Metode Numerik

   Metode numerik adalah teknik yang melibatkan data angka. Metode ini disebut juga dengan nama metode *heuristic*. Metode numerik digunakan karena efektif dan efisien dalam penyelesaian soal matematis pada *hardware* dan *software* komputer. Alasan lain mengapa orang lebih memilih metode numerik adalah karena metode analitik tidak mampu menyelesaikan soal matematis pada *software* yang dinilai terlalu kompleks. Dalam analisis numerik, diterima tidaknya hasil aproksimaksi didasarkan pada toleransi pada galat yang telah disepakati sebeumnya. Galat yang semakin kecil artinya semakin bagus formula yang digunakan (Zakaria & Muharramah, 2023).

2. Polinomial

   Adalah sebuah fungsi yang memiliki derajat terntentu. Misalnya fungsi polinomial derajat 2 adalah sebagai berikut
   $$f(x) = x^2 + x + 1$$
3. Estimasi

   Adalah cara untuk mengetahui perkiraan nilai suatu hal tanpa harus menghitung nilai aslinya. Jadi hanya perkiraan saja. Hal ini sangat berguna untuk mengetahui nilai suatu hal dengan cepat tanpa harus menghitung dengan detail.

4. Turunan

   Turunan adalah pengembangan dari limit. Berikut ini definisi dari fungsi turunan
   $$f'(x) = \lim_{x \to 0} \frac{f(x+h) - f(x)}{h}$$
5. Deret Taylor

Newton Rapson adalah sebuah pengembangan dari deret taylor. Berikut ini adalah deret taylor. Jika $f: \mathbb{R} \longrightarrow \mathbb{R}$ adalah sebuah fungsi yang terdifferensiasi sebanyak n+1 kali. Maka $f(x)$ dapat ditulis dengan

$$f(x) = f(x_i) + f'(x_i)(x - x_i) + \frac{f''(x_i)}{2!}(x - x_i)^2 + \frac{f'''(x_i)}{3!}(x - x_i)^3 + \cdots = \frac{\sum_{n=0}^{\infty} f^{(n)}(x_i)}{n!}$$

6. Konvergensi

   Adalah nilai yang mendekati suatu fungsi atau parameter tertentu. Hal ini diperlukan untuk mengetahui apakah nilai yang diestimasikan sudah mendekati nilai aslinya atau belum.

7. Teorema Rolle

   Jika suatu fungsi kontinu di interval tertutup dan terdifferensiasi pada interval terbuka dan nilai fungsi pada ujung-ujung interval sama, maka setidaknya ada satu titik di dalam interval dimana turunan fungsi itu nol.

8. Teorema Nilai Rata-rata

   Jika suatu fungsi kontinu pada interval tertutup dan terdifferensiasi pada interval terbuka. Maka setidaknya ada satu titik di dlam interval dimana turunan fungsi tersebut sama dengan rata-rata tingkat pertumbuhan fungsi di seluruh interval.

**Materi Inti**

1. Metode Newton Raphson

   Metode ini adalah pengembangan dari deret taylor. Berikut ini adalah metode Newton Raphson:

$$f(x) \approx f(x_k) + (x - x_k)f'(x_k) + \frac{1}{2!}(x - x_k)^2 f''(x_k) + \cdots + \frac{1}{n!}(x - x_k)^n f^{(n)}(x_k)$$

- Ambil dua kalimat rumus yang pertama dari seluruh rumus di atas

$$f(x) \approx f(x_k) + (x - x_k)f'(x_k)$$

- Asumsikan $x = x_{k+1}$ adalah Solusi dari persamaan $f(x) = 0$

$$0 = f(x_k) + (x_{k+1} - x_k)f'(x_k)$$

- Pindah ruas

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Berikut ini algoritma dari metode Newton Raphson:

```python
def newton_raphson_poly(X, y, beta, max_iters=100, tolerance=1e-6):
    for _ in range(max_iters):
        gradient, hessian = gradient_hessian_poly(X, y, beta)
        beta_new = beta - np.linalg.solve(hessian, gradient)
        if np.linalg.norm(beta_new - beta) < tolerance:
            break
        beta = beta_new
```

```
    return beta

beta_initial = np.zeros(X_poly.shape[1])
beta_optimal = newton_raphson_poly(X_poly, y_train, beta_initial)
```

2. Root Mean Squared Error(RMSE)

RMSE adalah cara mengukur indicator di model regresi. RMSE mnegukur perbedaan nilai yang diprediksi dengan nilai sebenarnya. Berikut ini rumus RMSE:

$$\sqrt{\frac{1}{W}\sum_{i=2}^{N} w_i {u_i}^2}$$

Dimana, $W$ adalah total berat populasi, $N$ adalah banyaknya observasi, $w_i$ adalah berat observasi ke-i dan $u_i$ adalah galat pada observasi ke-i.

3. R-Squared

Adalah seberapa besar pengaruh variabel independen terhadap variabel dependen. Adapun rumus dari R-Squared adalah:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{\sum_{i=1}^{n}(y_i - \bar{y})}$$

Dimana, $y_i$ adalah nilai yang diamati, $\hat{y}_i$ adalah nilai yang diprediksi dan $\bar{y}$ adalah rata-rata dari nilai yang diamati

**Penerapan**

1. Regresi Linier

Metode Newton Raphson dapat digunakan untuk mengestimasikan nilai dari regresi linier atau akar-akar dari persamaan regresi linier. Bentuk regresi linier pada observasi ke-$n$ adalah sebagai berikut:

$$y_i = \beta_0 x_i + \beta_i x_i + \varepsilon_i, \qquad i = 1,2,\dots,n$$

Dimana $\varepsilon$ adalah error-nya(Pesce & Wiley, 2008). Berikut ini adalah algoritma yang digunakan untuk menghitung regresi linier:

```
from sklearn.linier_model import LinierRegression
from sklearn.metrics import mean_squared_error, r2_score

regression_model = LinierRegression()
regression_model.fit(X_train, y_train)
y_predicted = regression_model.predict(X_test)
```

2. Regresi Polinomial (Regresi Non-Linier)

Dalam kasus tertentu, regresi linier tidak bisa digunakan, maka dari itu regresi non-linier akan digunakan untuk menyelesaikan masalah ini (Nocedal & Wright, 2000). Pada artikel ini regresi non-linier yang akan digunakan adalah regresi polinomial dengan derajat 2. Berikut ini rumus regresi polinomial dengan derajat 2:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

Berikut ini adalah algoritma yang digunakan untuk menghitung regresi polinomial dengan derajat 2:

```
poly_reg = PolynomialFeatures(degree=2)
X_poly = poly_reg.fit_transform(X_train)
```

3. Matriks Jacobi

Matriks ini berisi turunan parsial pertama. Matriks ini dilambangkan $\boldsymbol{J}$. Matriks ini berukuran $m \times n$ dengan elemen matrik ke-$(i, j)$ adalah $\boldsymbol{J}_{ij} = \frac{\partial f_i}{\partial x_j}$ atau ditulis juga,

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Dari matriks jacobi ini, yang awalnya penggunaannya hanya dengan untuk turunan parsial petama, akan dikembangkan untuk digunakan pada turunan parsial kedua.

4. Matriks Hessian

Matriks Hessian adalah fungsi multivariat $f(x, y, z, \ldots)$, yang dapat ditulis juga dengan $H(f)$, $Hf$ atau, $H_f$, yang menyatukan turunan parsial kedua ke dalam matriks. Berikut ini bentuk matriks hessian,

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} & \cdots \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} & \cdots \\ \frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Matriks Hessian digunakan untuk perhitungan turunan parsial kedua, berikut ini adalah algoritma yang digunakan:

```
def gradient_hessian_poly(X, y, theta):
    m = len(y)
    predictions = X.dot(theta)
    residuals = predictions - y
    gradient = (2/m) * X.T.dot(residuals)
    hessian = (2/m) * X.T.dot(X)
    return gradient, hessian
```

## HASIL

Implementasi artikel ini adalah pada data harga rumah yang akan diaplikasikan regresi linier dan non-linier serta regresi non-linier dengan metode Newton Raphson. Dalam artikel ini penerapan akan membutuhkan hal-hal di bawah ini:

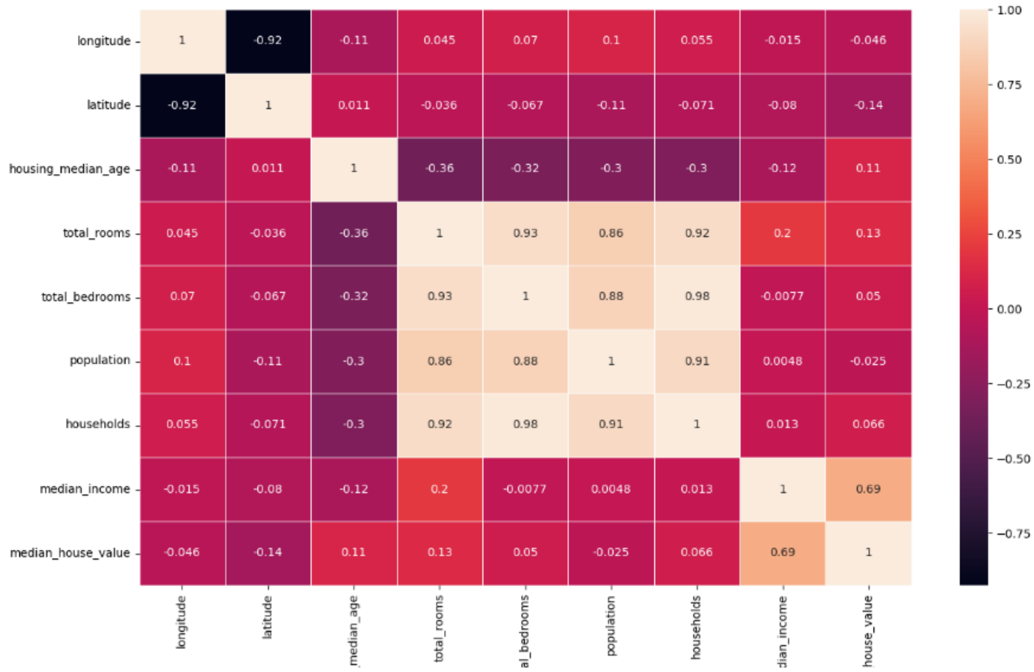
1. Python
   Untuk memudahkan dalam perhitungan regresi non-linier yaitu regresi polinomial dengan metode newton Raphson maka akan digunakan Python beserta *library* yang telah tersedia di Python. Berikut ini *library* yang dibutuhkan: numpy, pandas, matplotlib, seaborn, scikit-learn.
2. Dataset Harga Rumah
   Data yang digunakan adalah data harga rumah yang didapat dari website Kaggle. Model regresi non-linier juga didapat dari website Kaggle (Alok, 2019). Di dalam website tersebut dijelaskan bahwa pada data ini telah digunakan regresi linier dan regresi non-linier(regresi polynomial). Disini hasil dari model regresi non-linier ini akan diaplikasikan dengan metode Newton Raphson. Berikut ini adalah contoh dari datanya:

**Gambar 1. Data Harga Rumah**

| longitude | latitude | housing_median_age | total_rooms | total_bedrooms | population | households | median_income | median_house_value | ocean_proximity |
|---|---|---|---|---|---|---|---|---|---|
| -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | 322.0 | 126.0 | 8.3252 | 452600.0 | NEAR BAY |
| -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | 2401.0 | 1138.0 | 8.3014 | 358500.0 | NEAR BAY |
| -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | 496.0 | 177.0 | 7.2574 | 352100.0 | NEAR BAY |
| -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | 558.0 | 219.0 | 5.6431 | 341300.0 | NEAR BAY |
| -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | 565.0 | 259.0 | 3.8462 | 342200.0 | NEAR BAY |

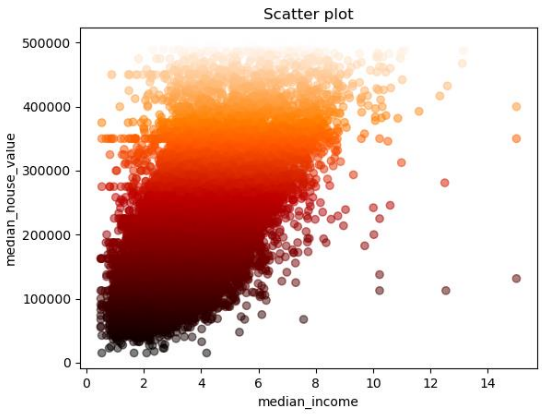
Lalu sekarang akan kita implementasikan pada data di atas. Sebelum dilakukan regresi, akan kita lihat dahulu koefiesien korelasi antar variabel. Lalu, akan dipilih data yang memiliki koefisien korelasi yang tertinggi. Berikuti ini perhitungan dari koefisien korelasi:

**Gambar 2. Koefisien Korelasi antar Variabel**



Dari data tersebut, koefisien korelasi tertinggi adalah antara variabel *median_house_value* dan *median_income* dengan nilai koefisien korelasi sebesar 0.69. Berikut ini visualisasi dari *median_house_value* dan *median_income*.

**Gambar 3. Visualisasi Data median_income dan median_house_value**




Sebelum diaplikasikan regresi linier, data akan dibagi menjadi 2 yaitu *train* 80% dan *test* 20%. Berikut ini cara mengaplikasikannya

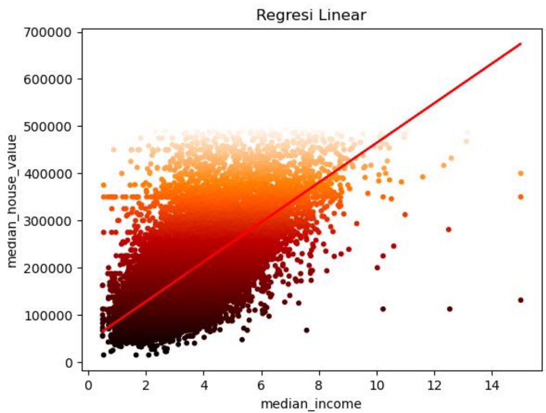
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 
0)
```

Sebelum mengarah ke metode Newton Raphson, akan ditunjukkan hasil regresi yang telah ada di dalam website Kaggle(Alok, 2019). Setelah itu akan ditunjukkan hasil estimasi dan perbandingan dari semua metode yang ada.

1. Pemodelan dengan Regresi Linier

   Berikut ini adalah visualisasi regresi linier:

**Gambar 4. Regresi Linier pada Data median_income dan median_house_value**



Pemodelan dengan regresi linier menghasilkan nilai error sebagai berikut:

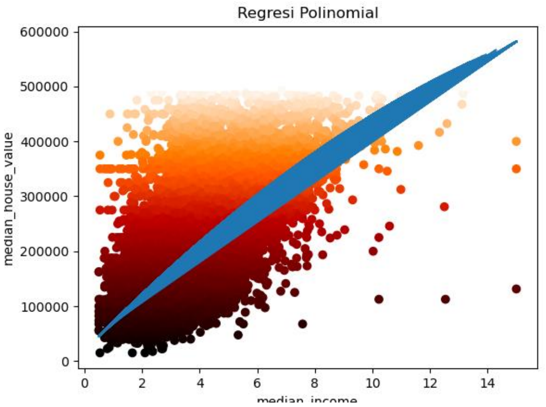
**Tabel 1. Error pada Regresi Linier**

| ERROR | NILAI |
|---|---|
| **RMSE** | 84941.05152406936 |
| **R-SQUARED** | 0.4466846804895944 |

2. Pemodelan dengan Regresi Polinomial

Berikut ini adalah visualisasi regresi linier di dalam website Kaggle(Alok, 2019):

**Gambar 5. Regresi Polinomial derajat 2 pada Data median_income dan median_house_value**




Pemodelan dengan regresi polinomial menghasilkan nilai error sebagai berikut:

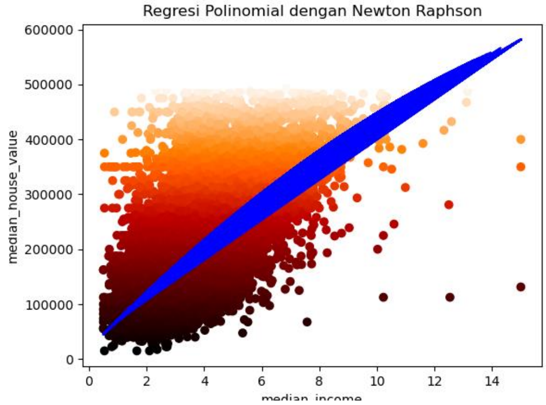
**Tabel 2. Error pada Regresi Polinomial Derajat 2**

| ERROR | NILAI |
|---|---|
| **RMSE** | 84699.90676455047 |
| **R-SQUARED** | 0.44982190770645936 |

3. Pemodelan dengan Regresi Polinomial dengan metode Newton Raphson

Berikut ini adalah visualisasi regresi polinomial dengan metode Newton Raphson pada Gambar 6:

**Gambar 6. Regresi Polinomial Derajat 2 dengan Metode Newton Raphson pada Data median_income dan median_house_value**




Rumus umum dari regresi linier derajat 2 adalah:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

Dengan metode newton Raphson ditemukan akar-akar dari polinomial derajat 2 yaitu **53878.50556201** dan **-1089.15005102**.

Pemodelan dengan regresi polinomial menghasilkan nilai error sebagai berikut:

**Tabel 3. Error pada Regresi Polinomial Derajat 2 dengan Metode Newton Raphson**

| ERROR | NILAI |
|---|---|
| **RMSE** | 84699.90676455054 |
| **R-SQUARED** | 0.44982190770645836 |

Perbandingan hasil error dengan masing-masing model adalah:

**Tabel 4. Perbandingan Error pada Masing-masing Model**

| | REGRESI LINIER | REGRESI POLINOMIAL | REGRESI POLINOMIAL DENGAN NEWTON RAPHSON |
|---|---|---|---|
| **RMSE** | 84941.05152406936 | 84699.90676455047 | 84699.90676455054 |
| **R-SQUARED** | 0.4466846804895944 | 0.44982190770645936 | 0.44982190770645836 |

## KESIMPULAN DAN SARAN

Hasil estimasi dari regresi polynomial derajat 2 dengan metode Newton Raphson adalah 53878.50556201 dan -1089.15005102. Error dari hasil metode Newton Raphson adalah RMSE 84699.90676455054 dan r-squared 0.44982190770645836. Hasil ini sangat mendekati dengan nilai dari metode regresi polinomial biasa. Selisih RMSE regresi polinomial dengan metode Newton Raphson dengan regresi polinomial biasa adalah 0.00000000007275957614. Sedangkan, selisih r-squarednya adalah 0.00000000000000099920. Terlihat metode Newton Raphson memiliki nilai galat yang rendah. Sehingga, kesimpulan dari artikel ini adalah **metode Newton Raphson sangat efektif dalam menentukan estimasi nilai-nilai akar dari persamaan regresi polinomial.**

Dari hasil regresi linier, regresi polynomial dan regresi polynomial dengan metode Newton Raphson, perbedaan error dari ketiga model tidak terlalu signifikan. Model yang terbaik untuk data harga rumah adalah regresi linier dengan RMSE yang lebih kecil sebesar 84941.05152406936 dan dengan r-squared sebesar 0.4466846804895944. Saran untuk penelitisn berikutnya mungkin bisa menggunakan metode selain metode Newton Raphson.

## DAFTAR PUSTAKA

Ali, Z., Rafaat, Z., & Alkaki, A. (2020). *Estimation of Non-Linear Regression Parameters by Newton Raphson's Method of Nonlinear Equations*. https://doi.org/10.13140/RG.2.2.16973.15842

Alok. (2019). *Non Linear regression*. https://www.kaggle.com/code/alokevil/non-linear-regression/input

Gireesh Kumar, D., Venkata Sireesha, N., Ganesh, A., Kotb, H., Aboras, K. M., Zeinoddini-Meymand, H., & Kamel, S. (2023). Design of an Optimized Asymmetric Multilevel Inverter with Reduced Components Using Newton-Raphson Method and Particle Swarm Optimization. *Mathematical Problems in Engineering*, *2023*. https://doi.org/10.1155/2023/9966708

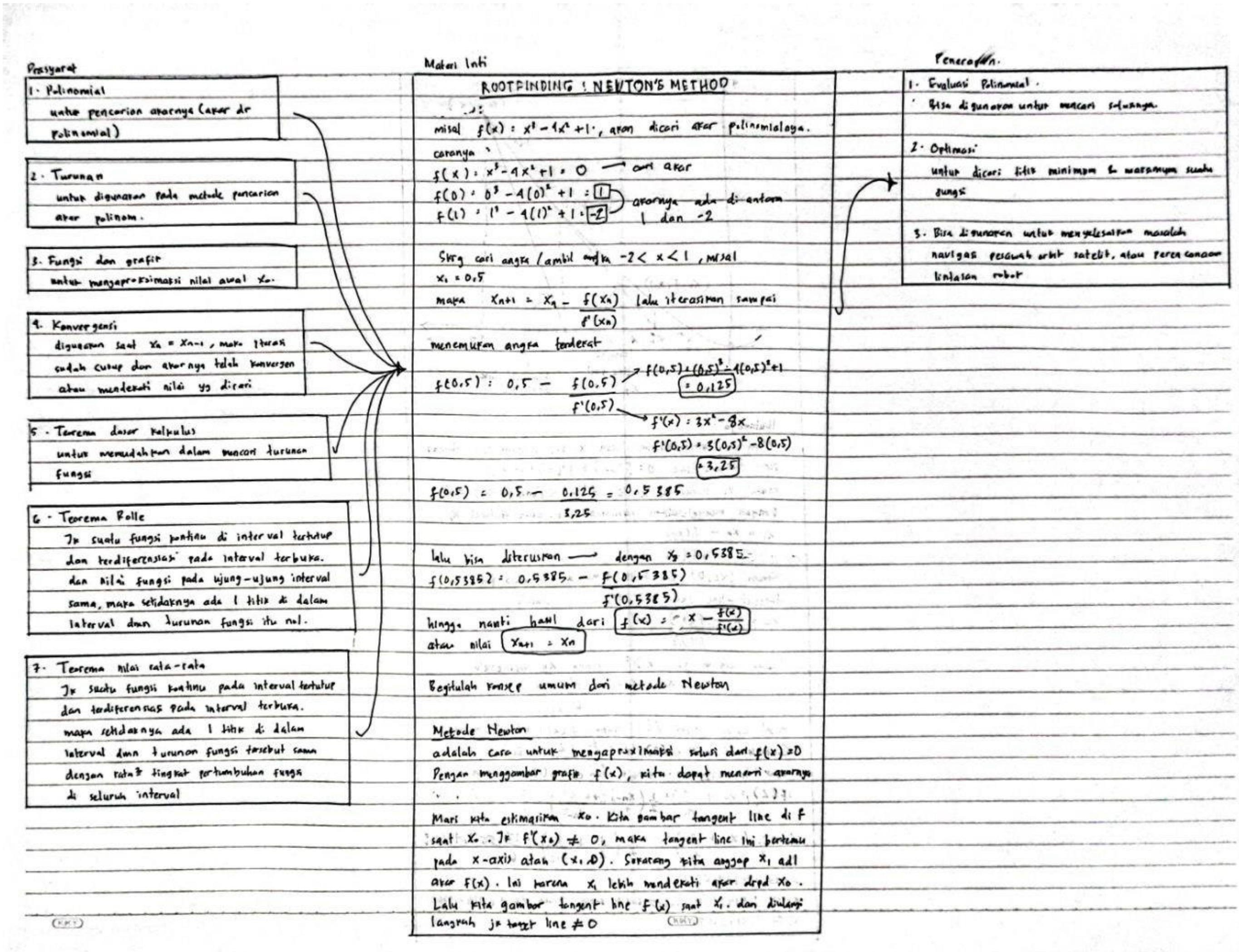
Nocedal, J., & Wright, S. J. (2000). *Numerical Optimization Second Edition*. https://www.math.uci.edu/~qnie/Publications/NumericalOptimization.pdf

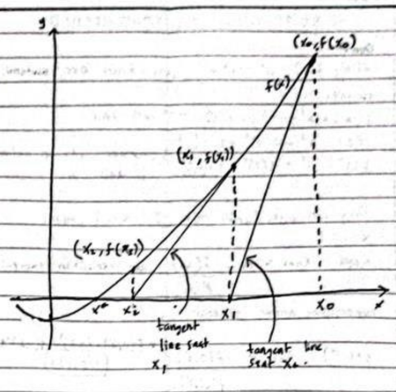
Otkun, Ö. (2021). Newton–Raphson based scalar speed control and optimization of IM. *Automatika*, *62*(1), 55–64. https://doi.org/10.1080/00051144.2020.1846322

Pesce, W. J., & Wiley, P. B. (2008). *Linear Models in Statistics* (2nd ed.). https://www.utstat.toronto.edu/~brunner/books/LinearModelsInStatistics.pdf

Zakaria, L., & Muharramah, U. (2023). *Pengantar Metode Numerik(Solusi Masalah dengan Matematika)*. Aura. http://repository.lppm.unila.ac.id/50807/1/Pengantar%20Metode%20Numerik%20%283%29.pdf

## LAMPIRAN


Prasyarat
1. Polinomial
untuk pencarian akarnya (akar dr polinomial)
2. Turunan
untuk digunakan pada metode pencarian akar polinom.
3. Fungsi dan grafik
untuk mengaproksimasi nilai awal x0.
4. Konvergensi
digunakan saat xn = xn-1, maka iterasi sudah cukup dan akarnya telah konvergen atau mendekati nilai yg dicari
5. Teorema dasar kalkulus
untuk memudahkan dalam mencari turunan fungsi
6. Teorema Rolle
Jk suatu fungsi kontinu di interval tertutup dan terdiferensiasi pada interval terbuka. dan nilai fungsi pada ujung-ujung interval sama, maka setidaknya ada 1 titik di dalam interval dan turunan fungsi itu nol.
7. Teorema nilai rata-rata
Jk suatu fungsi kontinu pada interval tertutup dan terdiferensias pada interval terbuka. maka setidaknya ada 1 titik di dalam interval dan turunan fungsi tersebut sama dengan rata² tingkat pertumbuhan fungsi di seluruh interval
Materi Inti
ROOTFINDING : NEWTON'S METHOD
misal f(x) : x³ − 4x² + 1, akan dicari akar polinomialnya.
caranya :
f(x) : x³ − 4x² + 1 = 0 → cari akar
f(0) : 0³ − 4(0)² + 1 : 1
f(1) : 1³ − 4(1)² + 1 : −2
akarnya ada di antara 1 dan −2
Skrg cari angka / ambil angka −2 < x < 1, misal
x1 = 0,5
maka xn+1 = xn − f(xn)/f'(xn) lalu iterasikan sampai menemukan angka terdekat
f(0,5) : 0,5 − f(0,5)/f'(0,5)
f(0,5) = (0,5)³ − 4(0,5)² + 1 = 0,125
f'(x) : 3x² − 8x
f'(0,5) = 3(0,5)² − 8(0,5) = 3,25
f(0,5) = 0,5 − 0,125/3,25 = 0,5385
lalu bisa diteruskan → dengan x2 = 0,5385
f(0,5385) : 0,5385 − f(0,5385)/f'(0,5385)
hingga nanti hasil dari f(x) = x − f(x)/f'(x)
atau nilai xn+1 = xn
Begitulah konsep umum dari metode Newton
Metode Newton
adalah cara untuk mengaproximasi solusi dari f(x) = 0
Dengan menggambar grafik f(x), kita dapat mencari akarnya
Mari kita estimasikan x0. Kita gambar tangent line di f saat x0. Jk f'(x0) ≠ 0, maka tangent line ini bertemu pada x-axis atau (x1, 0). Sekarang kita anggap x1 adl akar f(x). Ini karena x1 lebih mendekati akar drpd x0. Lalu kita gambar tangent line f(x) saat x1. dan diulangi langkah jk tangent line ≠ 0
Penerapan.
1. Evaluasi Polinomial.
Bisa digunakan untuk mencari solusinya.
2. Optimasi
untuk dicari titik minimum & maksimum suatu fungsi
3. Bisa digunakan untuk menyelesaikan masalah navigasi pesawat orbit satelit, atau perencanaan lintasan robot




Maka kita aproksimasikan, dgn $x_0$ sbg awalnya dan dengan pers. tangent line $y = f(x_0) + f'(x_0)(x - x_0)$.

maka $x_1$ harus memenuhi $f(x_0) + f'(x_0)(x - x_0) = 0$

Dengan menyelesaikan persamaan tsb. akan didapat $x_1$,

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Karena $(x_2, 0)$ adl titik temu di x-axis pada tangent-line saat $x_1$. Maka $x_2$ adalah

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

Secara umum jk $n > 0$, maka $x_n$ memenuhi

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

misal akan dicari $\sqrt{2}$, maka $f(x) = x^2 - 2$, maka

$$x_n = x_{n-1} - \frac{x_{n-1}^2 - 2}{2x_{n-1}} = \frac{1}{2} x_{n-1} + \frac{1}{x_{n-1}}$$

$$x_1 = \frac{1}{2}\left(x_0 + \frac{2}{x_0}\right) = \frac{1}{2}\left(2 + \frac{2}{2}\right) = \frac{3}{2} = 1{,}5$$

$$x_2 = \frac{1}{2}\left(x_1 + \frac{2}{x_1}\right) = \frac{1}{2}\left(\frac{3}{2} + \frac{2}{3/2}\right) = \frac{17}{6} \approx 1{,}416666$$

$x_3 \approx 1{,}414215686$

$x_4 \approx 1{,}414213562$ sudah sama maka $\sqrt{2} = 1{,}414213562$

$x_5 \approx 1{,}414213562$

Prasyarat

1. Polinomial
   untuk pencarian akarnya (akar dr polinomial)

2. Turunan
   untuk digunakan pada metode pencarian akar polinom.

3. Fungsi dan grafik
   untuk mengaproksimasi nilai awal $x_0$.

4. Konvergensi
   digunakan saat $x_n = x_{n-1}$, maka iterasi sudah cukup dan akarnya telah konvergen atau mendekati nilai yg dicari

5. Teorema dasar kalkulus
   untuk memudahkan dalam mencari turunan fungsi

6. Teorema Rolle
   Jk suatu fungsi kontinu di interval tertutup dan terdiferensiasi pada interval terbuka. dan nilai fungsi pada ujung-ujung interval sama, maka setidaknya ada 1 titik di dalam interval dan turunan fungsi itu nol.

7. Teorema nilai rata-rata
   Jk suatu fungsi kontinu pada interval tertutup dan terdiferensias pada interval terbuka. maka setidaknya ada 1 titik di dalam interval dan turunan fungsi tersebut sama dengan rata² tingkat pertumbuhan fungsi di seluruh interval

Materi In

misal

caranya

$f(x) =$

$f(0) =$

$f(1)$

Skrg

$x_1 = 0,$

maka

menemu-

$f(0,5$

$f(0,5)$

lalu bis-

$f(0,5385$

hingga

atau nil

Begitulah

Metode

adalah

Pengan

Mari k

saat $x_0$

pada x

akar $f(x$

Lalu k

langkah

Materi Inti

## ROOTFINDING : NEWTON'S METHOD

misal $f(x) = x^3 - 4x^2 + 1$, akan dicari akar polinomialnya.

caranya:

$f(x) = x^3 - 4x^2 + 1 = 0$ → cari akar

$f(0) = 0^3 - 4(0)^2 + 1 = \boxed{1}$

$f(1) = 1^3 - 4(1)^2 + 1 = \boxed{-2}$

akarnya ada di antara 1 dan -2

Skrg cari angka / ambil angka $-2 < x < 1$, misal

$x_1 = 0,5$

maka $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$ lalu iterasikan sampai menemukan angka terdekat

$f(0,5) = 0,5 - \dfrac{f(0,5)}{f'(0,5)}$

$f(0,5) = (0,5)^3 - 4(0,5)^2 + 1$ $\boxed{= 0,125}$

$f'(x) = 3x^2 - 8x$

$f'(0,5) = 3(0,5)^2 - 8(0,5)$ $\boxed{= 3,25}$

$f(0,5) = 0,5 - \dfrac{0,125}{3,25} = 0,5385$

lalu bisa diteruskan → dengan $x_2 = 0,5385$

$f(0,5385) = 0,5385 - \dfrac{f(0,5385)}{f'(0,5385)}$

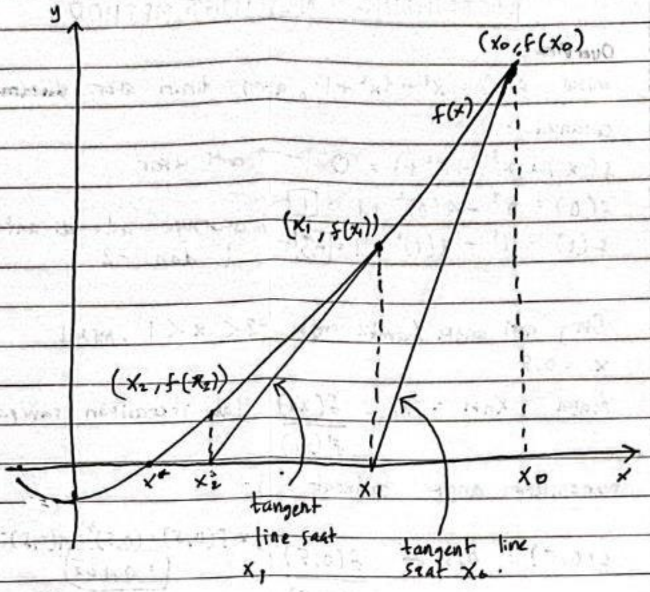
hingga nanti hasil dari $f(x) = x - \dfrac{f(x)}{f'(x)}$ atau nilai $x_{n+1} = x_n$

Begitulah konsep umum dari metode Newton

Metode Newton

adalah cara untuk mengaproximasi solusi dari $f(x) = 0$. Dengan menggambar grafik $f(x)$, kita dapat mencari akarnya.

Mari kita estimasikan $x_0$. Kita gambar tangent line di f saat $x_0$. Jk $f'(x_0) \neq 0$, maka tangent line ini bertemu pada x-axis atau $(x_1, 0)$. Sekarang kita anggap $x_1$ adl akar $f(x)$. Ini karena $x_1$ lebih mendekati akar drpd $x_0$. Lalu kita gambar tangent line $f(x)$ saat $x_1$ dan diulangi langkah jk tangent line $\neq 0$




Maka kita aproksimasikan, dgn $x_0$ sbg awalnya dan dengan pers. tangent line $y = f(x_0) + f'(x_0)(x - x_0)$.

maka $x_1$ harus memenuhi $f(x_0) + f'(x_0)(x - x_0) = 0$

Dengan menyelesaikan persamaan tsb. akan didapat $x_1$,

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Karena $(x_2, 0)$ adl titik temu di x-akis pada tangent-line saat $x_1$. Maka $x_2$ adalah

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

Secara umum jk $n > 0$, maka $x_n$ memenuhi

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

misal akan dicari $\sqrt{2}$, maka $f(x) = x^2 - 2$, maka

$$x_n = x_{n-1} - \frac{x_{n-1}^2 - 2}{2x_{n-1}} = \frac{1}{2}x_{n-1} + \frac{1}{x_{n-1}}$$

$$x_1 = \frac{1}{2}\left(x_0 + \frac{2}{x_0}\right) = \frac{1}{2}\left(2 + \frac{2}{2}\right) = \frac{3}{2} = 1,5$$

$$x_2 = \frac{1}{2}\left(x_1 + \frac{2}{x_1}\right) = \frac{1}{2}\left(\frac{3}{2} + \frac{2}{3/2}\right) = \frac{17}{6} \approx 1,416666$$

$x_3 \approx 1,414215686$

$x_4 \approx 1,414213562$ sudah sama maka $\sqrt{2} = 1,414213562$

$x_5 \approx 1,414213562$

KIKY

Penerapan.

1. Evaluasi Polinomial.
   Bisa digunakan untuk mencari solusinya.

2. Optimasi
   untuk dicari titik minimum & maksimum suatu fungsi

3. Bisa digunakan untuk menyelesaikan masalah navigasi pesawat orbit satelit, atau perencanaan lintasan robot

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
data = pd.read_csv("./housing.csv")
data.head()

fig, ax = plt.subplots(figsize=(20, 5))  # Adjust the size as needed

# Hide the axes
ax.xaxis.set_visible(False)
ax.yaxis.set_visible(False)
ax.set_frame_on(False)

# Create the table
table = ax.table(cellText=data.head().values, colLabels=data.columns, cellLoc='center',
loc='center')

# Adjust the layout
plt.tight_layout()

# Save the table as a picture
plt.savefig("data_head.png", bbox_inches='tight', pad_inches=0.1)

# Pearson correlation
plt.subplots(figsize=(15, 9))
cor = data.corr()
sns.heatmap(cor, annot=True, linewidths=.5)
plt.savefig('Koef Korelasi.png')
plt.show()

# taking two variables
data =
data.drop(["housing_median_age","households","total_bedrooms","longitude","latitude","tot
al_rooms","population","ocean_proximity"], axis=1)
data.head()

# Visualisasi data
X = data.drop("median_house_value", axis=1)
y = data["median_house_value"]
```

```
plt.scatter(X, y, alpha=0.5,c=y, cmap='gist_heat')
plt.title('Scatter plot')
plt.xlabel('median_income')
plt.ylabel('median_house_value')
plt.savefig('data.png')
plt.show()

# Split data
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state =
0)

# Regresi Linier
# Predict
y_predicted = regression_model.predict(X_test)

# model evaluation
rmse = np.sqrt(mean_squared_error(y_test, y_predicted))
r2 = r2_score(y_test, y_predicted)

# printing values
print('Slope:' ,regression_model.coef_)
print('Intercept:', regression_model.intercept_)
print('Root mean squared error: ', rmse)
print('R2 score: ', r2)

# data points
plt.scatter(X_train, y_train, s=10, c=y_train, cmap='gist_heat')
plt.title('Regresi Linear')
plt.xlabel('median_income')
plt.ylabel('median_house_value')

# predicted values
plt.plot(X_test, y_predicted, color='r')
plt.savefig('linear_regression.png')
plt.show()

# residual plot
residual = y_test - y_predicted
sns.residplot(residual,y_predicted, lowess=True,scatter_kws={'alpha':
0.5},line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.show()
```

```
# regresi polinomial
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state =
0)
poly_reg = PolynomialFeatures(degree=2)
X_poly = poly_reg.fit_transform(X_train)
pol_reg = LinearRegression()
pol_reg.fit(X_poly, y_train)
def viz_polymonial():
    plt.scatter(X_train, y_train, c=y_train, cmap='gist_heat')
    plt.title('Regresi Polinomial')
    plt.plot(X_train, pol_reg.predict(poly_reg.fit_transform(X_train)))
    plt.xlabel('median_income')
    plt.ylabel('median_house_value')
    plt.savefig('regresiPolynom.png')
    plt.show()
    return
viz_polymonial()

# Predict
X_p = poly_reg.fit_transform(X_test)
y_predicted = pol_reg.predict(X_p)

# model evaluation
rmse = np.sqrt(mean_squared_error(y_test, y_predicted))
r2 = r2_score(y_test, y_predicted)

# printing values
print('Slope:' ,regression_model.coef_)
print('Intercept:', regression_model.intercept_)
print('Root mean squared error: ', rmse)
print('R2 score: ', r2)

# residual plot
res = y_test - y_predicted
sns.residplot(res,y_predicted, lowess=True,scatter_kws={'alpha': 0.5},line_kws={'color':
'red', 'lw': 1, 'alpha': 0.8})
plt.show()

# metode newton raphson
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
```

```python
from sklearn.metrics import mean_squared_error, r2_score
import seaborn as sns
from sklearn.model_selection import train_test_split

# Splitting data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Generating polynomial features
poly_reg = PolynomialFeatures(degree=2)
X_poly = poly_reg.fit_transform(X_train)

def gradient_hessian_poly(X, y, beta):
    m = len(y)
    predictions = X.dot(beta)
    residuals = predictions - y
    gradient = (2/m) * X.T.dot(residuals)
    hessian = (2/m) * X.T.dot(X)
    return gradient, hessian

def newton_raphson_poly(X, y, beta, max_iters=100, tolerance=1e-6):
    for _ in range(max_iters):
        gradient, hessian = gradient_hessian_poly(X, y, beta)
        beta_new = beta - np.linalg.solve(hessian, gradient)
        if np.linalg.norm(beta_new - beta) < tolerance:
            break
        beta = beta_new
    return beta

# Initial guess for beta (zeros)
beta_initial = np.zeros(X_poly.shape[1])

# Optimize using Newton-Raphson
beta_optimal = newton_raphson_poly(X_poly, y_train, beta_initial)

def viz_polynomial():
    plt.scatter(X_train, y_train, c=y_train, cmap='gist_heat')
    plt.plot(X_train, poly_reg.fit_transform(X_train).dot(beta_optimal), color='blue')
    plt.title('Regresi Polinomial dengan optimasi Newton Raphson')
    plt.xlabel('median_income')
    plt.ylabel('median_house_value')
    plt.savefig('Regresi_polinom_dengen_optimasi_newton_raphson.png')
    plt.show()

viz_polynomial()
```

```
# Predict
X_test_poly = poly_reg.fit_transform(X_test)
y_predicted = X_test_poly.dot(beta_optimal)

# Model evaluation
rmse = np.sqrt(mean_squared_error(y_test, y_predicted))
r2 = r2_score(y_test, y_predicted)

# Printing values
print('Coefficients:', beta_optimal[1:])  # Excludes the intercept
print('Intercept:', beta_optimal[0])  # Intercept term
print('Root mean squared error:', rmse)
print('R2 score:', r2)

# Residual plot
residuals = y_test - y_predicted
sns.residplot(x=y_predicted, y=residuals, lowess=True, scatter_kws={'alpha': 0.5},
line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel('Predicted values')
plt.ylabel('Residuals')
plt.show()
```