

# *Interpolasi dan Pencocokan Kurva*

*Fadjar Fathurrahman*

## *1 Polinomial Interpolasi Newton*

**Soal 1.** Implementasikan fungsi atau subroutin dalam Python untuk implementasi algoritma pada Gambar 18.7 untuk implementasi polinomial Newton. Uji hasil yang Anda dapatkan dengan menggunakan data-data yang diberikan pada contoh 18.2 (polinomial kuadrat) dan 18.3 (polinomial kubik) pada Chapra. Lengkapi jawaban Anda dengan membuat plot seperti pada Gambar 18.4 dan 18.6 pada Chapra.

**Soal 2.** Gunakan data pada soal Chapra 18.5 untuk mengevaluasi nilai  $f(x) = \ln(x)$  pada  $x = 2$  dengan menggunakan polinomial kubik. Coba variasikan titik-titik yang digunakan (*base points*) dan perhatikan nilai estimasi kesalahan yang diberikan oleh fungsi/subrutin yang sudah Anda buat pada soal sebelumnya.

## *2 Polinomial Interpolasi Lagrange*

**Soal 3.** Implementasikan fungsi atau subroutin dalam Python untuk implementasi algoritma pada Gambar 18.11 untuk implementasi polinomial Lagrange. Uji hasil yang Anda dapatkan dengan menggunakan data-data yang diberikan pada contoh 18.2 (polinomial kuadrat) dan 18.3 (polinomial kubik) pada Chapra.

**Soal 4.** Kerjakan Contoh 18.7 pada Chapra dengan menggunakan fungsi/subrutin interpolasi Lagrange yang sudah Anda buat pada soal sebelumnya sehingga Anda dapat mereproduksi Gambar 18.12 pada Chapra. Pada Contoh 18.17 Anda diminta untuk mengestimasi kecepatan penerjun pada  $t = 10$  s, yang berada di antara dua titik data terakhir. Untuk polinom orde-1, gunakan dua titik data terakhir, untuk orde-2 gunakan tiga titik data terakhir, dan seterusnya sampai orde-4.

## *3 Interpolasi Spline Kubik*

**Soal 5.** Implementasikan fungsi atau subroutin dalam Python untuk implementasi algoritma pada Gambar 18.18 untuk implementasi spline kubik

(natural). Uji hasil yang Anda dapatkan dengan menggunakan data-data yang diberikan pada contoh 18.10.

#### 4 Regresi Linear

**Soal 6.** Implementasikan fungsi atau subroutin dalam Python untuk implementasi algoritma pada Gambar 17.6 untuk implementasi regresi linear. Uji dengan menggunakan data pada Contoh 17.1.

#### 5 Regresi Linear (Notasi Matriks-Vektor)

Tinjau model linear berikut:

$$y = f(x; w_0, w_1) = w_0 + w_1 x \quad (1)$$

di mana  $w_1$  (atau kemiringan atau *slope*) dan  $w_0$  (titik potong sumbu  $y$  atau *intercept*) adalah parameter model. Misalnya, diberikan suatu nilai atau input  $x_n$ , kita dapat menghitung output dari model sebagai:

$$y_n = f(x_n; w_0, w_1) = w_0 + w_1 x_n \quad (2)$$

Definisikan:

$$\mathbf{x}_n = \begin{bmatrix} 1 \\ x_n \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \quad (3)$$

Dengan menggunakan notasi ini, model linear pada Persamaan (2) dapat ditulis menjadi:

$$y_n = f(x_n; w_0, w_1) = \mathbf{w}^T \mathbf{x}_n \quad (4)$$

Dalam bentuk matriks-vektor, dapat dituliskan sebagai berikut:

$$\mathbf{y} = \mathbf{X}\mathbf{w} \quad (5)$$

dengan  $\mathbf{X}$  adalah matriks input:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix} \quad (6)$$

dan  $\mathbf{y}$  adalah vektor kolom:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \quad (7)$$

Diberikan himpunan pasangan data  $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$  kita ingin mencari parameter  $\mathbf{w}$  yang meminimumkan rata-rata kesalahan kuadrat  $\mathcal{L}$  yang didefinisikan sebagai:

$$\mathcal{L} \equiv \frac{1}{N} \sum_{n=1}^N (y_n - \mathbf{w}^T \mathbf{x}_n)^2 \quad (8)$$

Dapat ditunjukkan bahwa parameter yang membuat  $\mathcal{L}$  menjadi minimum adalah

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (9)$$

**Soal 7.** Buat fungsi/subrutin dalam Python untuk membuat matriks  $\mathbf{X}$  dengan argumen input  $\mathbf{x}$  dan  $\mathbf{y}$  dan menghitung  $\mathbf{w}$  berdasarkan persamaan (9). Uji fungsi yang sudah Anda buat dengan menggunakan data pada Contoh 17.1 dan bandingkan parameter *slope* dan *intercept* yang Anda dapatkan pada soal ini dan soal sebelumnya. Anda dapat menggunakan `np.linalg.inv` untuk menghitung invers matriks atau `np.linalg.linalg` jika Anda mengubah permasalahan ini menjadi sistem persamaan linear.

Kita juga dapat menggunakan Persamaan (9) untuk model linear<sup>1</sup> polinom lebih tinggi dari satu. Misalkan, pada model polinom kuadrat:

$$y = w_0 + w_1 x + w_2 x^2 \quad (10)$$

matriks  $\mathbf{X}$  menjadi:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{bmatrix} \quad (11)$$

dan vektor  $\mathbf{w}$  menjadi:

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} \quad (12)$$

Formula yang sama juga dapat digunakan untuk model multilinear dengan dua variabel independen  $x^{(1)}$  dan  $x^{(2)}$  (tanda <sup>(1)</sup> dan <sup>(2)</sup>

<sup>1</sup> Model linear didefinisikan sebagai model yang parameternya linear (berpangkat satu)

bukan menyatakan pangkat, namun variabel yang berbeda)<sup>2</sup>:

<sup>2</sup> Silakan gunakan variabel lain seperti  $t, x, y, z$

$$y = w_0 + w_1x^{(1)} + w_2x^{(2)} \quad (13)$$

di mana sekarang matrix  $\mathbf{X}$  menjadi:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^{(1)} & x_1^{(2)} \\ 1 & x_2^{(1)} & x_2^{(2)} \\ \vdots & \vdots & \vdots \\ 1 & x_N^{(1)} & x_N^{(2)} \end{bmatrix} \quad (14)$$

**Soal 8.** Kembangkan fungsi/subrutin Python yang sudah Anda buat sehingga dapat menerima argumen opsional  $m$ , di mana  $m \geq 1$  adalah orde polinomial yang ingin digunakan. Uji dengan menggunakan data pada Contoh 17.5 di buku Chapra.

**Soal 9.** Modifikasi fungsi/subrutin Python sudah Anda buat untuk regresi linear sehingga dapat digunakan untuk regresi multilinear dan uji fungsi yang Anda buat dengan menggunakan data pada Contoh 17.6 di buku Chapra.

## 6 Soal Tambahan

### Soal 10. Chapra Latihan 18.5

Diberikan data berikut:

$x$	1.6	2	2.5	3.2	4	4.5
$f(x)$	2	8	14	15	8	2

- Hitung  $f(2.8)$  dengan menggunakan polinomial interpolasi Newton dengan orde 1 sampai 3.
- Gunakan Pers. (18.18) pada Chapra untuk mengestimasi kesalahan untuk setiap prediksi.

**Soal 11. Chapra Latihan 18.11** Gunakan interpolasi invers dengan menggunakan polinomial interpolasi kubik dan metode bagi dua (*bisection*) untuk menentukan nilai  $x$  yang memenuhi  $f(x) = 0.23$  untuk data dalam tabel berikut.

$x$	2	3	4	5	6	7
$y$	0.5	0.333	0.25	0.2	0.1667	1.1429

**Soal 12. Chapra Latihan 18.26** Fungsi Runge dapat dinyatakan sebagai berikut:

$$f(x) = \frac{1}{1 + 25x^2}$$

- (a) Buat plot dari fungsi tersebut interval dari  $x = -1$  sampai  $x = 1$ .
- (b) Buat polinomial interpolasi Lagrange orde 4 dengan menggunakan nilai fungsi yang disampel secara seragam:  $x = -1, -0.5, 0, 0.5, 1$ .  
Buat juga plot dari polinomial tersebut. Gunakan untuk menghitung nilai  $f(0.8)$ .
- (c) Ulangi bagian sebelumnya dengan menggunakan polinom orde 5 sampai 10.

**Soal 13. Chapra Latihan 18.27** Fungsi *humps* dapat ditulis sebagai berikut.

$$f(x) = \frac{1}{(x - 0.3)^2 + 0.01} + \frac{1}{(x - 0.9)^2 + 0.04} - 6$$

Hitung nilai fungsi ini pada titik-titik dalam interval  $x = 0$  sampai  $x = 1$  dengan jarak antar titik 0.1. Gunakan interpolasi spline kubik pada data yang Anda hasilkan dan buat plot dari fungsi *humps* dengan hasil interpolan spline yang Anda dapatkan.

**Soal 14. Chapra Latihan 17.4** Gunakan regresi kuadrat terkecil untuk mencocokkan garis lurus ke data berikut.

$x$	6	7	11	15	17	21	23	29	29	37	39
$x$	29	21	29	14	21	15	7	7	13	0	3

Plot data dan garis lurus (persamaan linear) yang Anda dapatkan dalam satu plot. Hitung juga koefisien korelasi dan determinasi.

**Soal 15. Chapra Latihan 17.6** Gunakan regresi kuadrat terkecil untuk mencocokkan polinomial orde-1 dan orde-2

$x$	1	2	3	4	5	6	7	8	9
$y$	1	1.5	2	3	4	5	8	10	13

**Soal 16. Chapra Latihan 17.8**

Lakukan pencocokan model pangkat:

$$y = ax^b$$

pada data berikut.

$x$	2.5	3.5	5	6	7.5	10	12.5	15	17.5	20
$y$	13	11	8.5	8.2	7	6.2	5.2	4.8	4.6	4.3

Gunakan persamaan yang diperoleh untuk memprediksi keluaran model jika  $x = 9$ . Jelaskan transformasi apa yang harus Anda buat untuk mengubah permasalahan ini menjadi permasalahan regresi linear. Plot data dan kurva yang dihasilkan pada koordinat  $x - y$  dan koordinat di mana model menjadi linear (Lihat Gambar 17.9).

### Soal 17. Chapra Latihan 17.9

Lakukan pencocokan model eksponensial:

$$y = \alpha_1 e^{\beta_1 x}$$

pada data berikut.

$x$	0.4	0.8	1.2	1.6	2	2.3
$y$	800	975	1500	1950	2900	3600

Jelaskan transformasi apa yang harus Anda buat untuk mengubah permasalahan ini menjadi permasalahan regresi linear. Plot data dan kurva yang dihasilkan pada koordinat  $x - y$  dan koordinat di mana model menjadi linear (Lihat Gambar 17.9).

**Soal 18.** Titik simpul Chebyshev jenis pertama didefinisikan pada selang  $[-1, 1]$  dan dapat dituliskan sebagai berikut:

$$t_k = -\cos\left(\frac{k\pi}{n}\right), \quad k = 0, \dots, n \quad (15)$$

Gunakan titik simpul Chebyshev sebagai ganti dari titik-titik  $x$  yang memiliki jarak seragam untuk melakukan interpolasi dengan polinom orde 10 dan orde 20 untuk fungsi Runge pada interval  $[-1, 1]$ . Bandingkan hasilnya jika Anda menggunakan titik-titik  $x$  yang memiliki jarak seragam.

## 7 Kode Fortran

Pada bagian ini akan diberikan kode Fortran dari beberapa algoritma yang ada pada buku Chapra mengenai interpolasi.

Beberapa catatan mengenai Fortran:

- Kode Fortran disusun berdasarkan blok dengan kata kunci dan nama tertentu. Yang biasa kita gunakan adalah kata kunci PROGRAM dan SUBROUTINE. Akhir tiap blok ditandai dengan kata kunci END.
- Fortran dapat menebak tipe data yang kita gunakan berdasarkan nama variabel yang digunakan. Kita memaksa agar hal ini tidak

terjadi dengan menggunakan pernyataan `IMPLICIT NONE`, artinya seluruh variabel yang digunakan harus dideklarasikan terlebih dahulu beserta tipenya. Beberapa korespondensi tipe variabel:

- `REAL(8)` dengan `float64` pada Python
- `INTEGER` dengan `int`
- `LOGICAL` dengan `bool`
- Array pada Fortran dapat dimulai dari indeks bilangan bulat apa saja, namun secara default dimulai dari 1. Contoh:
  - `a(N)`: array `a` memiliki indeks 1 sampai `N`. Jumlah elemen array adalah `N`.
  - `a(0:N)`: array `a` memiliki indeks 0 sampai `N`. Jumlah elemen array adalah `N+1`.

Anda dapat menggunakan laman berikut jika pada komputer Anda tidak tersedia *compiler* untuk Fortran: [https://www.onlinegdb.com/online\\_fortran\\_compiler](https://www.onlinegdb.com/online_fortran_compiler)

### *Interpolasi Newton*

```

SUBROUTINE newton_interp(N, x, y, xi, yint, ea)
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: x(0:N)
  REAL(8) :: y(0:N)
  REAL(8) :: ea(0:N-1)
  REAL(8) :: yint(0:N)
  REAL(8) :: xi
  !
  REAL(8) :: fdd(0:N,0:N)
  INTEGER :: i, j, order
  REAL(8) :: yint2, xterm

  DO i = 0, n
    fdd(i,0) = y(i)
  END DO

  DO j = 1, n
    DO i = 0, n-j
      fdd(i,j) = ( fdd(i+1,j-1) - fdd(i,j-1) ) / ( x(i+j) -
        ↪ x(i) )
    END DO
  END DO

  xterm = 1.d0
  yint(0) = fdd(0,0)
  DO order = 1, N

```

```

    xterm = xterm * ( xi - x(order-1) )
    yint2 = yint(order-1) + fdd(0,order)*xterm
    ea(order-1) = yint2 - yint(order-1)
    yint(order) = yint2
END DO

RETURN
END SUBROUTINE

```

### Contoh 18.2

```

PROGRAM example_18_2
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: x(3), y(3)
  REAL(8) :: xi
  REAL(8), ALLOCATABLE :: ea(:)
  REAL(8), ALLOCATABLE :: yint(:)
  !
  REAL(8), ALLOCATABLE :: fdd(:, :)

  x = (/ 1.d0, 4.d0, 6.d0 /)
  y = (/ 0.d0, 1.386294d0, 1.791759d0 /)

  N = 2
  xi = 2.d0

  ALLOCATE( ea(0:N-1) )
  ALLOCATE( yint(0:N) )

  CALL newton_interp(N, x, y, xi, yint, ea)

  WRITE(*,*) 'yint = ', yint
  WRITE(*,*) 'ea   = ', ea

  DEALLOCATE( ea )
  DEALLOCATE( yint )

END PROGRAM

```

### Contoh 18.3

```

PROGRAM example_18_3
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: x(4), y(4)
  REAL(8) :: xi
  REAL(8), ALLOCATABLE :: ea(:)
  REAL(8), ALLOCATABLE :: yint(:)
  !
  REAL(8), ALLOCATABLE :: fdd(:, :)

  x = (/ 1.d0, 4.d0, 6.d0 , 5.d0 /)

```



```

y = (/ 0.d0, 1.386294d0, 1.791759d0, 1.609438d0 /)

N = 3
xi = 2.d0

ALLOCATE( ea(0:N-1) )
ALLOCATE( yint(0:N) )

CALL newton_interp(N, x, y, xi, yint, ea)

WRITE(*,*) 'yint = ', yint
WRITE(*,*) 'ea   = ', ea

WRITE(*,*) abs( yint(N) - log(xi) )

DEALLOCATE( ea )
DEALLOCATE( yint )

END PROGRAM

```

### *Interpolasi Lagrange*

```

SUBROUTINE lagrange_interp(N, x, y, xx, res)
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: x(0:N), y(0:N)
  REAL(8) :: xx, res
  !
  REAL(8) :: ss, pp
  INTEGER :: i, j

  ss = 0.d0
  DO i = 0, N
    pp = y(i)
    DO j = 0, N
      IF( i /= j ) THEN
        pp = pp*( xx - x(j) ) / ( x(i) - x(j) )
      END IF
    enddo
    ss = ss + pp
  END DO
  res = ss
  RETURN
END SUBROUTINE

```

### *Interpolasi Spline Kubik Natural*

```

SUBROUTINE interp_nat_cubic_spline( N, x, y, d2x, xu, yu, dy,
↳ d2y )
  !
  IMPLICIT NONE
  !
  INTEGER :: N

```

```

REAL(8) :: x(0:N), y(0:N)
REAL(8) :: d2x(0:N)
REAL(8) :: xu
REAL(8) :: yu
REAL(8) :: dy, d2y
!
INTEGER :: i
LOGICAL :: flag, is_in_interval
REAL(8) :: c1, c2, c3, c4, t1, t2, t3, t4

flag = .false.
i = 1
DO WHILE(.true.)
  !
  is_in_interval = (xu >= x(i-1)) .and. (xu <= x(i))
  !
  IF( is_in_interval ) THEN
    !
    !write(*,*) 'interval = ', i
    !
    c1 = d2x(i-1)/6.d0/( x(i) - x(i-1) )
    c2 = d2x(i)/6.d0/( x(i) - x(i-1) )
    c3 = y(i-1)/(x(i) - x(i-1)) - d2x(i-1)*(x(i) -
      ↪ x(i-1))/6.d0
    c4 = y(i)/(x(i) - x(i-1)) - d2x(i)*(x(i) - x(i-1))/6.d0
    !
    t1 = c1*( x(i) - xu )**3
    t2 = c2*( xu - x(i-1) )**3
    t3 = c3*( x(i) - xu )
    t4 = c4*( xu - x(i-1) )
    !
    yu = t1 + t2 + t3 + t4
    !
    t1 = -3.d0*c1*( x(i) - xu )**2
    t2 = 3.d0*c2*( xu - x(i-1) )**2
    t3 = -c3
    t4 = c4
    !
    dy = t1 + t2 + t3 + t4
    !
    t1 = 6.d0*c1*( x(i) - xu )
    t2 = 6.d0*c2*( xu - x(i-1) )
    !
    d2y = t1 + t2
    !
    flag = .true.
    !
  ELSE
    !
    i = i + 1
  END IF
  !
  IF( i == N + 1 .or. flag ) THEN
    EXIT ! break out of the loop
  END IF

```

```

END DO

IF( .not. flag ) THEN
  WRITE(*,*) "ERROR: xu is outside range of spline"
  RETURN
END IF

RETURN
END SUBROUTINE

SUBROUTINE decomp_trid(N, e, f, g)
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: e(N), f(N), g(N)
  INTEGER :: k
  DO k = 2,N
    e(k) = e(k)/f(k-1)
    f(k) = f(k) - e(k)*g(k-1)
  END DO
  RETURN
END SUBROUTINE

! should be called after calling decomp_trid
SUBROUTINE subs_trid(N, e, f, g, r, x)
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: e(N), f(N), g(N), r(N)
  REAL(8) :: x(N) ! output
  INTEGER :: k

  ! Forward subs
  DO k = 2,N
    r(k) = r(k) - e(k)*r(k-1)
  END DO

  ! back subs
  x(N) = r(N)/f(N)
  DO k = N-1,1,-1
    x(k) = ( r(k) - g(k)*x(k+1) ) / f(k)
  END DO
  RETURN
END SUBROUTINE

SUBROUTINE gen_trid_matrix(N, x, y, e, f, g, r)
  IMPLICIT NONE
  INTEGER :: N
  REAL(8) :: x(0:N), y(0:N)
  REAL(8) :: e(N-1), f(N-1), g(N-1), r(N-1)
  INTEGER :: i

  f(1) = 2.d0*( x(2) - x(0) )
  g(1) = x(2) - x(1)
  r(1) = 6.d0/( x(2) - x(1) ) * ( y(2) - y(1) )
  r(1) = r(1) + 6.d0/( x(1) - x(0) ) * ( y(0) - y(1) )

```

```

!
DO i = 2,N-2
  e(i) = x(i) - x(i-1)
  f(i) = 2.d0*( x(i+1) - x(i-1) )
  g(i) = x(i+1) - x(i)
  r(i) = 6.d0/( x(i+1) - x(i) ) * ( y(i+1) - y(i) )
  r(i) = r(i) + 6.d0/( x(i) - x(i-1) ) * ( y(i-1) - y(i) )
END DO
!
e(n-1) = x(n-1) - x(n-2)
f(n-1) = 2.d0*( x(n) - x(n-2) )
r(n-1) = 6.d0/( x(n) - x(n-1) ) * ( y(n) - y(n-1) )
r(n-1) = r(n-1) + 6.d0/( x(n-1) - x(n-2) ) * ( y(n-2) -
→ y(n-1) )

RETURN
END SUBROUTINE

```