Alternatif Pemodelan Persamaan Matematik dengan Metode Numerik

Kamal Mahmudi – 13507111[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]if17111@students.if.itb.ac.id

Dalam bidang rekayasa persamaan matematik harus dapat dimodelkan sehingga dapat dengan mudah diprogram. Pemodelan yang tepat, selain mempermudah pengguna, juga dapat mengoptimasi waktu eksekusi dan penggunaan sumber daya. Selain itu, pemodelan tertentu memungkinkan adanya fitur tambahan yang sering dibutuhkan seperti pembentukan fungsi turunan dan integral secara otomatis.

Makalah ini membahas bentuk alternatif pemodelan persamaan matematik dengan metode numerik yang mampu menangani pembentukan fungsi turunan dan integral secara otomatis bahkan pada derajat yang tinggi.

***Index interpolasi, turunan, integral, model matematik*.**

I. PENDAHULUAN

Pada umumnya, bahasa pemrograman memodelkan sebuah persamaan matematik sebagai sebuah metode yang menerima masukan nilai-nilai peubah yang terlibat dan menghasilkan keluaran dari persamaan matematik tersebut.

Hal ini memiliki banyak keterbatasan. Apabila seorang pengembang aplikasi menginkan bentuk turunan atau bentuk integral dari fungsi tersebut, pengembang harus memodelkan lagi secara manual bentuk turunan atau bentuk integral tersebut dalam metode yang terpisah. Bukan tidak mungkin fungsi asal mengalami perubahan seperti adanya perubahan konstanta yang mengharuskan perubahan model fungsi setiap metode yang telah dibuat sebelumnya. Keterbatasan ini terasa semakin menyulitkan ketika berhadapan dengan fungsi yang membutuhkan bentuk turunan atau bentuk integral pada level yang tinggi.

Keterbatasan lain pada pemodelan persamaan matematik secara biasa adalah ketika fungsi yang dimodelkan relatif rumit. Pemodelan bentuk turunan atau bentuk integralnya juga ikut menjadi rumit. Selain itu, eksekusi metode yang mewakilkan fungsi yang rumit tersebut akan berjalan lambat dan menghabiskan sumber daya yang seharusnya dapat dihindari.

Pengembangan model persamaan matematik dengan pendekatan metode numerik diharapkan mampu mengatasi beberapa keterbatasan fital tersebut.

II. METODE NUMERIK

Metode numerik adalah teknik yang digunakan untuk memformulasikan persoalan matematik sehingga dapat dipecahkan dengan operasi perhitungan / aritmatika biasa seperti tambah, kurang, kali, dan bagi.

Metode ini lahir karena keterbatasan metode analitik dalam menyelesaikan permasalahan yang sering muncul di dunia nyata di mana permasalahan tersebut sering melibatkan bentuk dan proses yang rumit.

Metode ini tidak menghasilkan solusi sejati, tetapi mendekati solusi tersebut dalam batas toleransi yang dapat disesuaikan.

A. Interpolasi Polinom

Dalam dunia praktek, tidak jarang ditemui data yang berbentuk tabel dan membutuhkan adanya pengolahan lanjut sehingga data tersebut dapat menjelaskan keterhubungan peubah-peubah yang terlibat.

Data dalam bentuk tabel tersebut bisa saja dihasilkan dari serangkaian percobaan yang dimaksudkan untuk menemukan fungsi yang menghubungkan peubah-peubah yang terlibat. Selain itu data dalam bentuk tabel juga dapat digunakan untuk mewakili fungsi yang sangat rumit sehingga pengguna data tidak perlu melakukan perhitungan ulang pada data yang sama.

Salah satu metode yang dapat digunakan untuk melakukan pencocokan titik data dengan sebuah kurva adalah metode interpolasi. Metode interpolasi in cocok digunakan pada data yang memiliki ketelitian yang tinggi sehingga kurva yang dihasilkan tepat melalui setiap titik yang diberikan.

Dasar dari pembentukan polinom interpolasi ini adalah interpolasi lanjar yang menginterpolasikan dua buah titik dengan sebuah garis lurus dengan persamaan garis. Tentunya dalam berbagai kasus, polinom yang terbentuk memiliki derajat yang lebih tinggi dan melibatkan banyak titik.

Polinom Newton merupakan polinom yang cukup popular, selain karena dukungannya terhadap banyak titik, polinom ini membentuk tabel selisih-terbagi satu kali yang dapat digunakan berkali-kali pada titik-titik yang berlainan. Selain itu penambahan titik baru pada data tidak merubah tabel selisih-terbagi sehingga polinom ini menjadi flexibel dan cocok digunakan pada kasus

derajat polinom yang tidak diketahui.

Polinom Newton dapat dinyatakan dalam hubungan rekursif sebagai:

(i) Rekurens: $p_n(x) = p_{n-1}(x) + a_n(x-x_0)(x-x_1)\ldots(x-x_{n-1})$

(ii) Basis: $p_0(x) = f(x_0) = a_0$

Dengan mengembangkan polinom tersebut hingga suku ke-n, dapat dimisalkan:

$a_0 = f(x_0)$

$a_1 = f[x_1, x_0]$

$a_2 = f[x_2, x_1, x_0]$

$a_n = f[x_n, x_{n-1}, \ldots, x_1, x_0]$

dimana a_0, a_1, a_2, …, a_n merupakan nilai selisih terbagi yang memenuhi:

$$f[xi, xj] = \frac{f(x_i) - f(x_j)}{x_i - x_j}$$

$$f[xi, xj, xk] = \frac{f[x_i,\ x_j] - f[x_j,\ x_k]}{x_i - x_k}$$

dan seterusnya.

Evaluasi polinom dapat dilakukan dengan melakukan subtitusi peubah *x* dengan nilai yang ingin dicari dan melakukan subtitusi x_0, x_1, …, x_n sesuai titik yang diberikan pada fungsi polinom.

B. Turunan Numerik

Turunan numerik adalah teknik analisis numerik untuk menghasilkan perkiraan turunan dari suatu persamaan matematik.

Terdapat tiga pendekatan dalam menghitung turunan numerik. Hampiran selisih maju (1) di mana titik lain yang berada setelah titik yang hendak dicari turunannya dijadikan acuan. Hampiran selisih mundur (2) di mana titik lain yang berada sebelum titik yang hendak dicari turunannya dijadikan acuan. Terakhir adalah pendekatan hampiran selisih pusat (3) di mana titik yang hendak dicari turunannya tidak dijadikan acuan, tetapi titik-titik sebelum dan sesudahnyalah yang digunakan.

$$f'(x) = \frac{f(x+h) - f(x)}{h} \tag{1}$$

$$f'(x) = \frac{f(x) - f(x-h)}{h} \tag{2}$$

$$f'(x) = \frac{f(x+h_1) - f(x-h_2)}{h_1 + h_2} \tag{3}$$

Pada data farik yang memiliki selisih titik yang sama dapat digunakan pendekatan lain seperti deret Taylor atau polinom interpolasi dalam menentukan turunan suatu fungsi bahkan hingga derajat yang tinggi.

C. Integrasi Numerik

Integrasi numerik adalah teknik analisis numerik untuk menghasilkan perkiraan integral tentu dari suatu persamaan matematik

$$I = \int_a^b f(x)\, dx$$

dimana *a* dan *b* diketahui dan *f*(*x*) adalah fungsi empirik yang diberikan secara eksplisit dalam bentuk persamaan atau secara empirik dalam bentuk tabel nilai.

Salah satu pendekatan dalam menurunkan rumus integrasi numerik adalah metode pias yang dengan tafsiran geometri integral tentu daerah integrasi dibagi atas sejumlah pias yang berbentuk segiempat. Luas daerah integrasi dihampiri dengan luas jumlah setiap pias.

Aturan titik tengah pada metode pias, menghampiri luas pias tersebut dengan cara mengambil titik tengah absis pias yaitu $x = x_0 + h/2$ dimana *h* merupakan lebar pias. Sehingga perkiraan nilai integrasi antara titik *a* dan *b* adalah:

$$\int_a^b f(x)\, dx \approx h \sum_{i=0}^{n-1} f_{i+1/2} \tag{4}$$

Pada data farik yang memiliki selisih titik yang sama dapat digunakan pendekatan lain seperti pemanfaatan ekstrapolasi Richardson.

.

III. PEMODELAN PERSAMAAN MATEMATIK

Saat ini dapat ditemukan beberapa pustaka pemrograman yang memungkinkan pemodelan suatu fungsi menjadi mudah dan dapat memanggil turunan atau integral dari fungsi tersebut secara otomatis. Salah satu pustaka yang popular adalah autodiff dimana yang juga mampu menangani fungsi dengan banyak peubah namun model ini membutuhkan fungsi yang telah terdefinisi. Pemodelan yang dibahas di sini adalah pemodelan dengan pemanfaatan metode numerik dimana model dapat dibangun dari fungsi yang terdefinisi atau dari kumpulan data farik.

A. Konsep Utama

Pemodelan persamaan matematik menggunakan konsep pemrograman berorientasi objek. Hal ini dimaksudkan agar baik fungsi asli maupun fungsi turunan atau fungsi integralnya memiliki kemampuan dan cara kerja yang sama karena berasal dari kelas yang sama. Karenanya kelas yang mewakili model persamaan matematik tersebut memiliki atribut *integral* dan *derivation* yang baru akan dibentuk / diciptakan ketika diperintahkan oleh pengguna.

Model tersebut harus dapat dibangun baik dengan memanfaatkan fungsi yang telah didefinisikan atau dari kumpulan data farik. Meskipun demikian, pada akhirnya model data yang dihasilkan merupakan tabel data selisih-terbagi polinom Newton, yang langsung dibuat di awal pembentukan objek sehingga pada saat dilakukan evaluasi atau prediksi nilai suatu titik, model telah siap untuk digunakan.

Alasan penggunaan polinom antara lain dikarenakan data yang dapat didukung oleh model tidak mengharuskan memiliki jarak yang sama, juga tidak mengharuskan berada pada jarak yang dekat. Dengan polinom, model dapat melakukan pembentukan fungsi turunan dengan mengambil jarak yang lebih kecil dari jarak data yang

disediakan, karena polinom dapat melakukan perkiraan nilai dari suatu titik. Begitu juga pada kasus pembentukan fungsi integral, model dapat menentukan lebar pias yang digunakan. Lebar pias ditentukan oleh suatu faktor pengali yang merupakan atribut dari kelas model persamaan matematik tersebut.

Pembentukan fungsi turunan dapat dilakukan secara rekursif untuk mencapai derajat tertentu. Titik-titik yang dijadikan model adalah sama dengan titik-titik yang dijadikan model pada fungsi awal. Turunan pada titik tersebut ditentukan dengan menggunakan pendekatan hampiran selisih pusat dengan menggunakan persamaan (3), kecuali ketika titik berada di awal (digunakan pendekatan hampiran selisih maju dengan menggunakan persamaan (1)) dan di akhir data (digunakan pendekatan hampiran selisih mundur dengan menggunakan persamaan (2)).

Pembentukan fungsi integral juga dapat dilakukan secara rekursif untuk mencapai derajat tertentu. Titik-titik yang dijadikan model juga sama dengan titik-titik yang dijadikan model pada fungsi awal. Nilai integral pada titik tersebut ditentukan dengan mengasumsikan nilai fungsi integral tersebut di titik awal bernilai 0 dan melakukan perhitungan numerik berdasarkan persamaan (4) dimana *a* merupakan salah satu titik yang terdaftar pada model dan *b* adalah titik berikutnya yang juga terdaftar pada model. Meskipun demikian terdapat atribut konstanta penambah pada atribut kelas model yang dapat diatur sehingga model fungsi tetap valid. Pendekatan yang digunakan dalam pembentukan fungsi integral adalah dengan metode pias di titik tengah.

B. Implementasi

Atribut model:

```
private Double cAdder;
public Double CAdder
{
  get
  {
    return cAdder;
  }
  set
  {
    cAdder = value;
  }
}
private Double hFactor;
public Double HFactor
{
  get
  {
    return hFactor;
  }
  set
  {
    hFactor = value;
  }
}
private List<List<Double>> data;
private Term integral;
private Term derivation;
```

Inisiasi model dengan masukan data farik:

```
public Term (List<List<Double>> newData)
{
  cAdder = 0;
  hFactor = 1;
  data = new List<List<Double>> ();

  data.Add (new List<Double> ());
  foreach (Double x in newData[0])
    data[0].Add (x);

  data.Add (new List<Double> ());
  foreach (Double y in newData[1])
    data[1].Add (y);

  int n = data[1].Count;
  for (int i = 0; i < n - 1; i++) {
    data.Add (new List<Double> ());
    for (int j = 0; j < n - i - 1; j++) {
      Double temp = data[i + 1][j + 1] - data[i +
1][j];
      temp /= (data[0][i + j + 1] - data[0][j]);
      data[i + 2].Add (temp);
    }
  }
}
```

Inisiasi model dengan masukan fungsi yang terdefinisi:

```
public delegate Double Function(Double x);

public Term(Function f, Double a, Double b, int n)
{
  cAdder = 0;
  hFactor = 1;
  data = new List<List<Double>> ();
  data.Add (new List<Double> ());
  data.Add (new List<Double> ());

  Double h = (b - a) / (n - 1);
  for(int i = 0; i < n; i++)
  {
    data[0].Add(a);
    data[1].Add(f(a));
    a += h;
  }

  for (int i = 0; i < n - 1; i++) {
    data.Add (new List<Double> ());
    for (int j = 0; j < n - i - 1; j++) {
      Double temp = data[i + 1][j + 1] - data[i +
1][j];
      temp /= (data[0][i + j + 1] - data[0][j]);
      data[i + 2].Add (temp);
    }
  }
}
```

Evaluasi / perkiraan nilai fungsi di titik tertentu:

```
public Double Evaluate (Double x)
{
  return Evaluate (x, data[1].Count - 1);
}

public Double Evaluate (Double x, int degrees)
{
  Double retval = data[1][0];
  Double diff = 1;
  for (int i = 0; i < degrees; i++) {
    diff *= (x - data[0][i]);
    retval += data[i + 2][0] * diff;
  }
  return retval + cAdder;
}
```

Membentuk fungsi integral:

```
private Double EvaluateIntegral (int i)
{
  Double retval = 0;
  int n = (int) (1 / hFactor);
  Double h = (data[0][i] - data[0][i - 1]) * hFactor;

  Double x = data[0][i - 1] + h / 2;
  retval += Evaluate(x);
  for(int j = 1; j < n; j++)
  {
    x += h;
    retval += Evaluate(x);
  }

  return retval * h;
}

public void BuildIntegral ()
{
  List<List<Double>> newData = new List<List<Double>>
();

  newData.Add (new List<Double> ());
  newData.Add (new List<Double> ());

  int n = data[0].Count;
  newData[0].Add(data[0][0]);
  newData[1].Add(0);
  Double total = 0;
  for(int i = 1; i < n; i++)
  {
    total += EvaluateIntegral(i);
    newData[0].Add(data[0][i])
```

```
    newData[1].Add(total);
  }

  integral = new Term(newData);
  integral.HFactor = hFactor;
}

public void BuildIntegral (int level)
{
  BuildIntegral ();
  if (level > 1)
    integral.BuildIntegral (level - 1);
}
```

Membentuk fungsi turunan:

```
public void BuildDerivation ()
{
  List<List<Double>> newData = new List<List<Double>>
();

  newData.Add (new List<Double> ());
  newData.Add (new List<Double> ());

  int i = 0, n = data[0].Count;
  Double dy, h1, h2;

  newData[0].Add (data[0][i]);
  h2 = (data[0][i + 1] - data[0][i]) * hFactor;
  dy = Evaluate(data[0][i] + h2) - data[1][i];
  newData[1].Add (dy / h2);

  for(i = 1; i < n - 1; i++)
  {
    newData[0].Add (data[0][i]);
    h1 = (data[0][i] - data[0][i - 1]) * hFactor;
    h2 = (data[0][i + 1] - data[0][i]) * hFactor;
    dy = Evaluate(data[0][i] + h2) -
Evaluate(data[0][i] - h1);
    newData[1].Add (dy / (h1 + h2));
  }

  newData[0].Add (data[0][i]);
  h1 = (data[0][i] - data[0][i - 1]) * hFactor;
  dy = data[1][i] - Evaluate(data[0][i] - h1);
  newData[1].Add (dy / h1);
  derivation = new Term(newData);
  derivation.HFactor = hFactor;
}

public void BuildDerivation (int level)
{
  BuildDerivation ();
  if (level > 1)
    derivation.BuildDerivation (level - 1);
}
```

Mengambil fungsi integral atau fungsi turunan:

```
public Term GetIntegral ()
{
  return GetIntegral (1);
}

public Term GetIntegral (int level)
{
  if (level > 1 && integral != null)
    return integral.GetIntegral (level - 1);

  return integral;
}

public Term GetDerivation ()
{
  return GetDerivation (1);
}

public Term GetDerivation (int level)
{
  if (level > 1 && derivation != null)
    return derivation.GetDerivation (level - 1);

  return derivation;
}
```

IV. HASIL DAN ANALISIS

Pengujian yang dilakukan dengan dua skenario, skenario pengujian pertama dilakukan untuk memodelkan persamaan sederhana dengan maksud untuk membandingkan hasil evaluasi model fungsi beserta model fungsi turunan dan model fungsi integralnya terhadap hasil yang sebenarnya untuk mengetahui tingkat akurasi pemodelan.

Skenario pengujian kedua dilakukan untuk memodelkan persamaan yang rumit dimana perbandingan evaluasi model turunan dan model integral terhadap nilai sebenarnya tidak dapat dilakukan. Pengujian ini dimaksudkan untuk melihat bahwa model tetap dapat memodelkan persamaan yang rumit sekalipun dan mampu berjalan seperti biasa.

A. Akurasi

Skenario pengujian pertama yang dilakukan adalah membuat sebuah objek model yang merupakan representasi fungsi $f(x) = x^8$ di sekitar titik $x = 0$ dan $x = 9$. Pemilihan fungsi ini untuk mengetahui akurasi dari model ketika membentuk fungsi turunan atau fungsi integral, karena fungsi tersebut mudah untuk diturunkan atau diintegrasikan secara manual untuk dibandingkan dengan hasil yang diperoleh model.

Pengujian dilakukan dengan membentuk turunan fungsi tersebut hingga derajat ke-8. Setiap fungsi turunan tersebut dievaluasi terhadap titik $x = 2.5$ dan dibandingkan dengan hasil yang diperoleh secara manual. Berikut adalah hasil yang diperoleh dengan menggunakan $h = 0.000001$:

	Penggunaan model	Nilai sebenarnya
$f(2.5)$	1525.87890625	1525.87890625
$f^{(1)}(2.5)$	4882.81186070	4882.81250000
$f^{(2)}(2.5)$	13671.86817155	13671.87500000
$f^{(3)}(2.5)$	32812.51547452	32812.50000000
$f^{(4)}(2.5)$	65625.03390837	65625.00000000
$f^{(5)}(2.5)$	104999.81824609	105000.00000000
$f^{(6)}(2.5)$	126000.00659290	126000.00000000
$f^{(7)}(2.5)$	100801.17214535	100800.00000000
$f^{(8)}(2.5)$	40318.04456209	40320.00000000

Tabel 1 Hasil evaluasi model fungsi dan turunannya

Turunan fungsi yang terakhir, yaitu $f(x) = 8!$, disimpan pada sebuah variabel lain dan dibangkitkan integral fungsi tersebut hingga derajat ke-8 yang mengembalikan fungsi tersebut ke fungsi asilnya yaitu $f(x) = x^8$. Berikut adalah hasil yang diperoleh dengan menggunakan $h = 0.000001$:

	Penggunaan model	Nilai sebenarnya
$f(2.5)$	40318.04456209	40320.00000000
$F^{(1)}(2.5)$	100877.86682630	100800.00000000
$F^{(2)}(2.5)$	126156.27772923	126000.00000000
$F^{(3)}(2.5)$	105162.68377131	105000.00000000
$F^{(4)}(2.5)$	65741.18077186	65625.00000000
$F^{(5)}(2.5)$	32876.05129312	32812.50000000
$F^{(6)}(2.5)$	13700.10927310	13671.87500000
$F^{(7)}(2.5)$	4893.33320377	4882.81250000
$F^{(8)}(2.5)$	1529.17252385	1525.87890625

Tabel 2 Hasil evaluasi model fungsi dan integralnya

Dari tabel 1 dan tabel 2 terlihat bahwa nilai evaluasi model fungsi tidak berbeda jauh dengan nilai sebenarnya. Perbedaan (galat) ini terus meningkat seiring dengan meningkatnya derajat turunan atau derajat integral yang dievaluasi. Hal ini disebabkan adanya akumulasi galat dari satu model fungsi ke model fungsi lain (integral atau turunannya). Meskipun demikian nilai h dapat diatur sedemikian rupa sehingga pada derajat turunan tertentu hasil evaluasi model fungsi yang diperoleh masih dapat diterima.

B. Kehandalan

Skenario pengujian berikutnya yang dilakukan adalah membuat sebuah objek model yang merupakan representasi fungsi $f(x) = \sqrt{x * e^x + \frac{x^3}{|1+\sin(x)|}}$ di sekitar titik $x = 0$ dan $x = 9$.

Hasil evaluasi model fungsi tersebut di titik $x = 2.5$ adalah 12.79424951. Hasil evaluasi model fungsi turunannya di titik yang sama dengan $h = 0.000001$ adalah 1.37098515.

Ketika model fungsi turunan disimpan pada variabel lain dan dari model tersebut dibangkitkan fungsi integralnya (yang tidak lain adalah fungsi aslinya) hasil evaluasi model yang diperoleh adalah 12.79362882 yang tidak berbeda jauh dengan hasil semula (selisih sekitar 0.0006). Hal ini menunjukkan model tetap mampu merepresentasikan persamaan matematik yang rumit.

V. KESIMPULAN

Suatu persamaan matematik dapat dihampiri dengan sebuah polinom interpolasi. Polinom interpolasi dapat dimodelkan secara matematik dengan kumpulan titik-titik dan nilai polinom pada titik tersebut.

Dalam membentuk model turunan dan model integralnya, polinom tersebut dapat digunakan untuk mengambil nilai persamaan matematik pada titik yang sangat dekat, sehingga tabel nilai yang dibutuhkan tidak perlu menampung data banyak karena akan dapat dilengkapi oleh polinom tersebut.

Pemodelan dengan polinom ini juga dapat diandalkan ketika menurunkan atau mengintegralkan fungsi ke derajat yang relatif tinggi ataupun pada pemodelan fungsi yang rumit untuk diturunkan atau diintegrasikan secara manual.

REFERENCES

Chapra, Seteven C dan Canale, Raymond, *Numerical Methods for Engineers with Personal Computer Applications*, MacGraw-Hills, Inc, 1992.

Hamilton, Said, dan Hamilton, Linda, *Calculus I: Differentiation and Integration*, Hamilton Education Guides, 2002.

Munir, Rinaldi, *Metode Numerik untuk Teknik Informatika*, 1997.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 13 Mei 2011

Kamal Mahmudi / 13507111