Seminar Nasional Royal (SENAR) 2018 ISSN 2622-9986 (cetak)
STMIK Royal – AMIK Royal, hlm. 61 – 66 ISSN 2622-6510 (online)
Kisaran, Asahan, Sumut - 3 September 2018

PEMROGRAMAN KOMPUTASI RUMUS EKSPLISIT METODE BEDA HINGGA UNTUK TURUNAN PERTAMA DENGAN MENGGUNAKAN MATLAB

Havid Syafwan[1], Mahdhivan Syafwan[2], William Ramdhan[3], Riki Andri Yusda[4]
[1]Manajemen Informatika, AMIK Royal Kisaran
[2]Matematika FMIPA, Universitas Andalas
[3,4]Teknik Komputer, AMIK Royal Kisaran
email: havid_syafwan@yahoo.com

Abstrak: *Metode beda hingga merupakan salah satu metode numerik yang paling populer dan mudah digunakan dalam menghitung hampiran turunan suatu fungsi. Rumus eksplisit dari koefisien beda hingga untuk turunan pertama dengan sebarang orde ketelitian telah diberikan oleh Khan dan Ohba berdasarkan observasi hasil numerik dan telah mereka buktikan secara matematis. Pada penelitian ini, rumus eksplisit beda hingga tersebut digunakan dalam membuat rancangan aplikasi komputasi untuk turunan fungsi dengan memakai bahasa pemrograman MATLAB (Matrix Laboratory). Pemrograman tersebut dapat memudahkan seseorang dalam menghitung turunan pertama dari sebarang fungsi, berapapun orde ketelitian yang diinginkan. Lebih lanjut, ketika galat dari turunan numerik dihitung di setiap orde ketelitian n, muncul fenomena osilasi pada kurva galat yang dihasilkan.*

Kata kunci: *turunan, rumus eksplisit, metode beda hingga, pemrograman MATLAB*

PENDAHULUAN

Persamaan diferensial merupakan topik kajian yang ada pada bidang matematika dan banyak digunakan dalam menyelesaikan berbagai masalah di bidang kimia, fisika, ekonomi, teknik elektro, industri, dan disiplin ilmu lainnya. Dalam hal ini persamaan diferensial seringkali dijadikan sebagai model matematika yang dapat mendeskripsikan suatu fenomena alam yang bersifat kompleks.

Untuk menyelesaikan suatu persamaan diferensial secara numerik, perlu dicari terlebih dahulu hampiran suku turunan yang muncul dalam persamaan diferensial tersebut. Salah satu metode numerik yang biasa digunakan untuk menghitung hampiran turunan suatu fungsi adalah metode beda hingga (*finite difference*). Pada metode ini domain fungsi dipartisi atas sejumlah titik dan rumus aproksimasi untuk turunan diperoleh dari ekspansi deret Taylor di satu atau lebih titik partisi (Mathews, 1992). Berdasarkan lokasi titik-titik partisi yang digunakan, metode beda hingga dibagi atas tiga jenis, yaitu beda maju (*forward difference*), beda mundur (*backward difference*) dan beda pusat (*central difference*).

Rumus umum beda hingga untuk turunan ke-*m* dengan ketelitian orde ke-*n* dapat dibangkitkan dengan suatu algoritma rekursif (Fornberg, 1988). Namun dalam implementasi perhitungannya, semakin tinggi tingkatan turunan fungsi dan orde ketelitian yang ingin dicari, maka semakin besar pula beban memori komputasi yang dibutuhkan. Hal ini karena semakin tinggi tingkatan turunan dan orde ketelitian, maka jumlah data (titik-titik partisi) yang diperlukan juga semakin banyak. Untuk mengatasi hal tersebut, diperlukan suatu rumus ekplisit beda hingga sehingga koefisien-koefisiennya dapat ditentukan secara langsung tanpa melewati proses perhitungan secara rekursif.

Rumus eksplisit dari koefisien-koefisien beda hingga untuk turunan ke-*m* dengan orde ketelian ke-*n* telah diformulasikan dalam (Khan dan Ohba, 1999) yang diperoleh berdasarkan observasi numerik. Pembuktian rumus eksplisit untuk turunan pertama pada jenis beda maju telah dibuktikan secara matematis (Khan dkk, 2003).

Pada penelitian ini rumus eksplisit tersebut akan digunakan dalam rancangan aplikasi pemrograman komputer sehingga seseorang dapat dengan mudah menghitung turunan pertama fungsi secara numerik, apapun fungsinya dan berapapun orde ketelitian yang diinginkan. Karena pemrograman ini menggunakan rumus ekplisit, maka komputasi yang dihasilkan menjadi sangat efektif dan efisien, meskipun dengan orde ketelitian yang sangat besar. Aplikasi komputasi ini dirancang

ISSN 2622-9986 (cetak)
ISSN 2622-6510 (online)

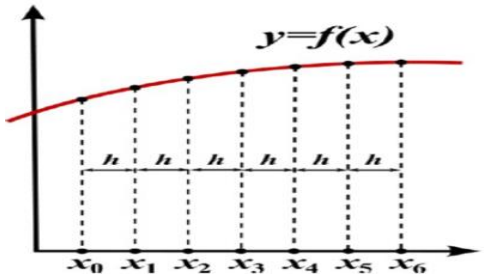
dengan menggunakan bahasa pemrograman MATLAB.

METODOLOGI

Metode beda hingga (*finite difference*) merupakan metode numerik yang digunakan untuk menghitung hampiran turunan dari suatu fungsi. Pada metode ini domain suatu fungsi $f(x)$ dipartisi atas sejumlah titik dan rumus aproksimasi untuk turunan diperoleh dari ekspansi deret Taylor di satu atau lebih titik partisi. Misalkan $[a,b]$ adalah domain $f(x)$ yang dipartisi atas $N + 1$ titik dengan lebar selang h, maka titik-titik partisinya adalah

$$x_i = a + ih;\ \ i = 0,1,\ldots, N. \tag{1}$$

Dalam hal ini $x_0 = a$ dan $x_N = b$. Selanjutnya nilai fungsi di masing-masing titik partisi ditulis $f_i=f(x_i)$. Contoh ilustrasi fungsi $f(x)$ yang domainnya dipartisi sebanyak 6 selang dapat dilihat pada Gambar 1.



Gambar 1. Ilustrasi fungsi $f(x)$ dengan domain dipartisi sebanyak 6 selang

Berdasarkan lokasi titik-titik partisi yang digunakan, terdapat tiga jenis metode beda hingga, yaitu beda maju (*forward difference*), beda mundur (*backward difference*), dan beda pusat (*central difference*) (Mathews, 1992).

Dalam (Khan dan Ohba, 1999) telah dikembangkan rumus eksplisit dari koefisien-koefisien beda hingga yang diformulasi berdasarkan deret Taylor. Pandang fungsi $f(x)$ dimana domainnya dipartisi atas titik-titik $x_k=x_0+kT$, $k=0,\pm1,\pm2,\ldots$ dengan T menyatakan lebar selang partisi. Deret Taylor dari $f(x)$ di sekitar $x=x_0$ diberikan oleh

$$f_k = f_0 + kTf_0^{(1)} + \frac{(kT)^2}{2!}f_0^{(2)} + \cdots + \frac{(kT)^n}{n!}f_0^{(n)} + \mathrm{O}(T^{n+1}), \tag{2}$$

dimana $f_k=f(x_k)$, $f_0^{(m)}=f^{(m)}(x_0)$ dengan m menyatakan tingkatan turunan, dan $\mathrm{O}(T^{n+1})$ merepresentasikan suku sisa (galat) sehingga deret Taylor di atas dikatakan memiliki orde keakuratan n.

Untuk hampiran turunan pertama suatu fungsi $f(x)$ di titik $x=x_0$, rumus ekplisitnya diberikan oleh

$$f_0^{(1)} \approx \frac{1}{T}\sum_k g_k f_k, \tag{3}$$

dimana koefisien g_k dan iterator k didefinisikan berdasarkan orde ketelitian dan jenis beda hingga sebagai berikut:

1. Beda maju dengan orde ketelitian n, rumus ekplisit koefisiennya diberikan oleh

$$g_{0,n}^{\mathrm{F},1} = -\sum_{j=1}^{n}(1/j) \tag{4}$$

dan

$$g_{k,n}^{\mathrm{F},1} = \frac{(-1)^{k+1}}{k}C_k^n, \quad k = 1,2,\ldots,n, \tag{5}$$

dimana C_b^a adalah $a!/(b!(a-b)!)$.

2. Beda mundur dengan orde ketelitian n, rumus ekplisit koefisiennya diberikan oleh

$$g_{0,n}^{\mathrm{B},1} = -g_{0,n}^{\mathrm{F},1} = \sum_{j=1}^{n}(1/j) \tag{6}$$

dan

$$g_{k,n}^{\mathrm{B},1} = -g_{k,n}^{\mathrm{F},1} = \frac{(-1)^{k}}{k}C_k^n, \quad k = 1,2,\ldots,n. \tag{7}$$

3. Beda pusat dengan orde ketelitian $2n$, rumus ekplisit koefisiennya diberikan oleh

$$g_{0,2n}^{\mathrm{C},1} = 0 \tag{8}$$

dan

$$g_{k,2n}^{\mathrm{C},1} = (-1)^{k+1}\frac{(n!)^2}{k(n-k)!(n+k)!} \tag{9}$$

dengan $k=\pm1,\pm2,\ldots \pm n$.

HASIL DAN PEMBAHASAN

Rancangan Penelitian

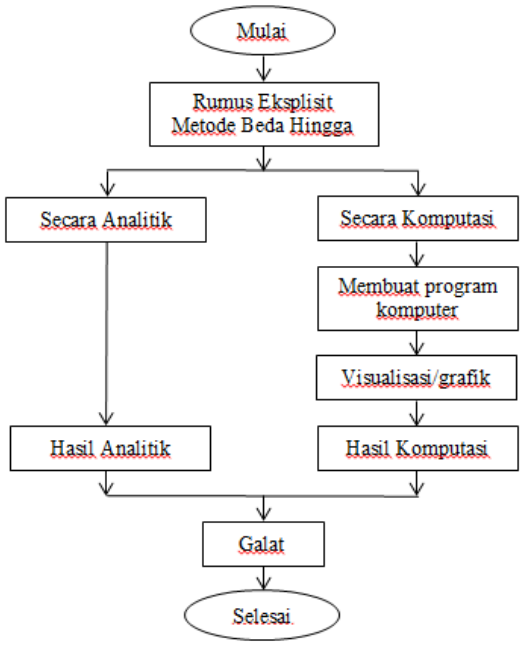
Langkah-langkah pelaksanaan penelitian adalah sebagai berikut:

a. Membahas formulasi rumus ekplisit metode beda hingga
b. Merancang struktur data
c. Penyusunan algoritma
d. Menerjemahkan algoritma ke dalam kode bahasa pemrograman

ISSN 2622-9986 (cetak)
ISSN 2622-6510 (online)

e. Menyusun kode tersebut menjadi sebuah program komputer
f. Menjalankan program
g. Menganalisis hasil visualisasi
h. Penulisan laporan

Diagram Alir Penelitian



Gambar 2. Diagram Alir Penelitian

Diagram Alir Program

Gambar 3. Diagram Alir Program

Pemrograman MATLAB

Pada pemrograman MATLAB, script file ditulis dalam format m-file yang merupakan kumpulan perintah yang ditulis menggunakan bahasa pemrograman MATLAB yang dapat disimpan dan dijalankan secara berulang. Caranya adalah dengan menyimpan script file dengan nama tanpa spasi dan dengan ekstensi .m.

Script dari program metode beda hingga diberikan dalam bentuk script sebagai berikut:

1. Script *function* yang memberikan rumus eksplisit dari koefisien beda hingga di satu titik x_0.

```
function [df0,g]=turunan(f,x0,n,h,t)
%Program ini mencari turunan pertama %dari fungsi f di x0 dengan orde %ketelian n dan lebar selang h dengan %menggunakan metode beda hingga dengan %jenis t, dimana t=1 (maju), t=2 %(mundur), t=3 (pusat).
%Untuk memanggil fungsi f, gunakan %sintaks berikut: @(x)f(x).
%Hasil turunan diberikan oleh df.
%Koefisien-koefisien diberikan oleh g.

if t==1 | t==2
   g0=0;
   for j=1:n
      g0=g0+1/j;
   end
   g0=-g0;

   for k=1:n
      g(k)=(-1)^(k+1)*factorial(n)...
                  /(k*factorial(n-k)...
                  *factorial(k));
   end

 if t==1
   xn=x0+n*h;
   x=[x0:h:xn]';

   y=0;
   for k=1:n
      y=y+g(k)*f(x(k+1));
   end
   df0=(g0*f(x(1))+y)/h;
   g=[g0 g];

end

if t==2
   g=-g;
   xn=x0-n*h;
```

ISSN 2622-9986 (cetak)
ISSN 2622-6510 (online)

```
   x=[x0:-h:xn]';
   y=0;
   for k=1:n
      y=y+g(k)*f(x(k+1));
   end
   df0=(-g0*f(x(1))+y)/h;
   g=[g(end:1) -g0];
end

end

if t==3
  if mod(n,2) ~= 0
     error('orde keakuratan beda pusat harus genap');
  end
  n=n/2;
   for k=-n:n
      if k==0
         g(n+1)=0;
      else
      g(k+n+1)=(-1)^(k+1)...
                  *factorial(n)^2 ...
                  /(k*factorial(n-k)...
                  *factorial(n+k));
      end
   end

   xn=x0+n*h;
   xm=x0-n*h;
   x=[xm:h:xn]';
   y=0;
%    M=2*n+1;
   for k=1:2*n+1
      y=y+g(k)*f(x(k));
   end
   df0=y/h;
end
```

2. Script *function* yang digunakan untuk memanggil fungsi turunan pada interval [a,b] dan memplot hasilnya.

```
function df=fungsiturunan(f,a,b,n,h,t)

x=[a:h:b]';
N=length(x);

for k=1:N
   df(k)=turunan(f,x(k),n,h,t);
end

plot(x,df,'bo-');
title('Plot Turunan Fungsi');
xlabel('x');
ylabel('df/dx');
xlim([a b]);
```

Contoh Kasus

Untuk lebih memahami penggunaan metode beda hingga pada program MATLAB, perhatikan contoh berikut.

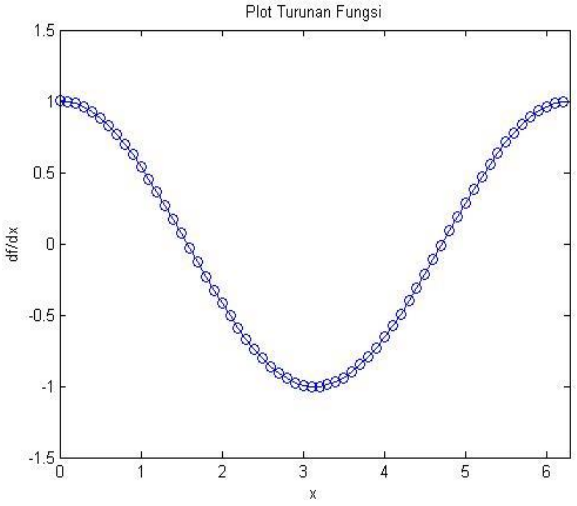
Tinjau fungsi $f(x) = \sin(x)$. Akan ditentukan hampiran turunan pertama dari $f(x)$ pada interval $[0,2\pi]$ dengan menggunakan beda pusat untuk orde ketelitian 2. Dalam hal ini digunakan lebar selang $T = 0.1$.

Penyelesaian.
Dari contoh di atas dapat diketahui :

- $f(x) = \sin(x)$
- metode beda pusat (t=3)
- selang [a,b]=[0,2*pi]
- orde ketelitian 2 (n=2)
- lebar selang 0.1 (h=0.1)

Dengan menggunakan nilai-nilai di atas untuk masing-masing variabel pada fungsi fungsiturunan, maka pada menu *command window* di MATLAB diketik script berikut:
fungsiturunan(@(x)sin(x),0,2*pi,2,0.1,3);

Plot fungsi turunan yang dihasilkan diberikan pada Gambar 4 yang menunjukkan kurva cosinus. Hal ini sesuai dengan turunan eksak dari sin(x).



Gambar 4. Grafik fungsi turunan $f(x) = \sin(x)$

Lebih lanjut, program turunan fungsi ini dapat digunakan untuk menghitung galat dari turunan numerik suatu fungsi di setiap orde ketelitian n. Misalkan untuk fungsi pada contoh sebelumnya, ingin dihitung galat dari turunan fungsi dari orde ketelitian *n=nawal* sampai

ISSN 2622-9986 (cetak)
ISSN 2622-6510 (online)

n=nakhir. Karena galat yang dihasilkan nantinya akan bernilai kecil, maka untuk melihat perbedaan nilainya dengan lebih jelas, didefinisikan galat untuk setiap *n* sebagai berikut:

$$galat = \log(\text{norm}(dfnum - dfeksak)), \quad (10)$$

dimana *dfnum* menyatakan fungsi turunan numerik dan *dfeksak* menyatakan fungsi turunan eksak.

Berikut script m-file untuk menghitung galat turunan numerik dari fungsi pada contoh sebelumnya untuk setiap orde ketelitian *n* dengan *nawal=2* dan *nakhir=100*. Karena digunakan metode beda pusat, maka orde ketelitian yang digunakan harus selalu bernilai genap. Oleh karena itu langkah yang dipakai adalah sebesar 2.

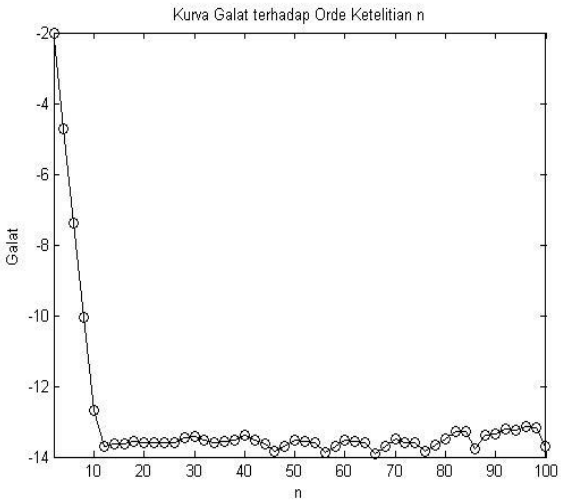
```
a=0;
b=2*pi;
h=0.1;
f=@(x)sin(x);
x=[a:h:b];
dfeksak=cos(x);
t=3;

nawal=2;
nakhir=100;

k=1;
for n=nawal:2:nakhir
   dfnum=fungsiturunan(f,a,b,n,h,t);
   galat(k)=log10(norm(dfeksak-dfnum));
   k=k+1;
end

plot([nawal:2:nakhir],galat,'ko-');
title('Kurva Galat terhadap Orde Ketelitian n');
xlabel('n');
ylabel('Galat');
xlim([nawal nakhir]);
```

Hasil yang diperoleh disajikan pada Gambar 5. Hal yang menarik pada gambar tersebut adalah munculnya fenomena galat yang mengalami osilasi. Untuk nilai *n* yang semakin membesar, nilai galat justru semakin naik. Hal ini kemungkinan disebabkan oleh kontribusi galat yang berasal dari pembulatan komputasi. Fenomena ini perlu dikaji dalam analisis numerik lebih lanjut.



Gambar 5. Galat turunan $f(x) = \sin(x)$ terhadap orde ketelitian n

SIMPULAN

Pada penelitian ini telah dirancang aplikasi komputasi untuk rumus eksplisit metode beda hingga pada turunan pertama suatu fungsi dalam bentuk program komputer menggunakan bahasa MATLAB. Program MATLAB ini akan memudahkan seseorang untuk menghitung turunan pertama fungsi secara numerik dan visualisasinya dalam bentuk grafik, apapun fungsinya dan berapapun orde ketelitian yang diinginkan.

Hal yang menarik adalah munculnya fenomena galat yang berosilasi terhadap orde ketelitian *n* yang dihasilkan dari metode beda hingga. Fenomena ini perlu dikaji dalam analisis numerik lebih lanjut.

UCAPAN TERIMAKASIH

Penulis mengucapkan terima kasih kepada Direktorat Riset dan Pengabdian Masyarakat, Kementrian Riset, Teknologi dan Pendidikan Tinggi Republik Indonesia yang telah mendanai penelitian ini melalui skema Penelitian Kerjasama Perguruan Tinggi dengan Kontrak No. 83/K1.1/LT.1/2018. Penulis juga mengucapkan terima kasih kepada AMIK Royal Kisaran dan Universitas Andalas atas dukungan dan fasilitas yang diberikan pada penelitian.

ISSN 2622-9986 (cetak)
ISSN 2622-6510 (online)

DAFTAR PUSTAKA

Fornberg, B. 1988. Generation of Finite Diference Formulas on Arbitrarily Spaced Grids. *Mathematics of Computation*. 51:184.

Khan, I. R dan R. Ohba. 1999. Closed form expressions for the finite diference approximations of first and higher derivatives based on Taylor series. *J. Comput. Appl. Math*. 107: 179-193.

Khan, I. R, R. Ohba, dan N. Hozumi. 2003. Mathematical proof of closed form expressions for finite diference approximations based on Taylor series. *J. Comput. Appl. Math*. 150: 303-309.

Mathews, J. H, K. D. Fink. 1992. *Numerical Methods for Computer Science, Engineering, and Mathematics.* Edisi ke-2. Prentice-Hall, Englewood Cliffs.