**JURTEKSI (Jurnal Teknologi dan Sistem Informasi)** ISSN 2407-1811 (Print)
Vol. VI No. 1, Des 2019, hlm. 65 - 70 ISSN 2550-0201 (Online)
DOI: https://doi.org/10.33330/jurteksi.v6i1.433
Available online at http://jurnal.stmikroyal.ac.id/index.php/jurteksi

# PENGEMBANGAN PROGRAM KOMPUTASI RUMUS EKSPLISIT METODE BEDA HINGGA PADA TURUNAN KEDUA MENGGUNAKAN MATLAB

**Havid Syafwan[1], Mahdhivan Syafwan[2*], William Ramdhan[3], Riki Andri Yusda[4]**
[1]Prodi D3 Manajemen Informatika STMIK Royal Kisaran
[2]Jurusan Matematika FMIPA Universitas Andalas
[3]Prodi D3 Teknik Komputer STMIK Royal Kisaran
[4]Prodi SI Sistem Komputer STMIK Royal Kisaran
*email: mahdhivan@sci.unand.ac.id

**Abstract:** In this paper we discuss the development of computational application program in MATLAB to calculate the second derivative of a function by using finite difference method for various types (forward, backward, or central) and any order of accuracy. The MATLAB program extends the similar one which has been developed for the first derivative. The program is designed as an implementation of the explicit formulas of finite difference formulated by Khan and Ohba. Through this program, one can easily calculate the second derivative of a function numerically with any order of accuracy and visualize it in a graphical form. Error of numerical derivatives in each order of accuracy confirm the validity of the obtained results.

**Keywords**: finite difference method, explicit formula, second derivative, MATLAB programming

**Abstrak:** Dalam makalah ini dibahas pengembangan program aplikasi komputasi pada MATLAB untuk menghitung turunan kedua suatu fungsi dengan menggunakan metode beda hingga pada berbagai tipe (maju, mundur, atau pusat) dan sebarang orde ketelitian. Program MATLAB ini melanjutkan program serupa yang telah dibuat untuk turunan pertama. Program ini dirancang sebagai implementasi dari rumus eksplisit beda hingga yang diformulasi oleh Khan dan Ohba. Melalui program ini, seseorang dengan mudah dapat menghitung turunan kedua suatu fungsi secara numerik dengan sebarang orde ketelitian dan memvisualisasinya dalam bentuk grafik. Galat dari turunan numerik di setiap orde ketelitian mengkonfirmasi kevalidan hasil yang diperoleh.

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

## PENDAHULUAN

Banyak sekali fenomena di alam ini yang dapat dimodelkan oleh persamaan diferensial. Untuk masalah-masalah yang lebih realistis, persamaan diferensial yang dihasilkan seringkali sulit untuk diselesaikan secara eksak, sehingga pendekatan numerik menjadi alternatif penyelesaian untuk persamaan diferensial tersebut.

Hal penting dalam penyelesaian persamaan diferensial secara numerik adalah menentukan hampiran turunan fungsi yang terlibat di dalamnya. Metode konvensional yang masih sering

**JURTEKSI (Jurnal Teknologi dan Sistem Informasi)** ISSN 2407-1811 (Print)
Vol. VI No. 1, Des 2019, hlm. 65 - 70 ISSN 2550-0201 (Online)
DOI: https://doi.org/10.33330/jurteksi.v6i1.433
Available online at http://jurnal.stmikroyal.ac.id/index.php/jurteksi

digunakan untuk menghitung hampiran turunan suatu fungsi adalah metode beda hingga (*finite difference*). Sebagaimana pada metode numerik lainnya, metode beda hingga mempartisi domain fungsi atas sejumlah titik. Selanjutnya rumus beda diperoleh dari ekspansi deret Taylor di satu atau lebih titik partisi [1]. Berdasarkan lokasi titik-titik partisi yang digunakan, metode beda hingga dibedakan 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 diperoleh dari suatu algoritma rekursif, artinya untuk memperoleh rumus turunan ke-*m* dengan ketelitian orde ke-*n*, mesti diketahui terlebih dahulu rumus turunan ke-(*m*-1) dengan ketelitian orde ke-(*n*-1). [2] telah mengembangkan algoritma rekursif tersebut yang darinya dapat dibuat tabel yang berisi koefisien-koefisien rumus beda maju, beda mundur dan beda pusat untuk beberapa tingkatan turunan fungsi dan orde ketelitian. Akan tetapi dalam implementasi perhitungannya, algoritma rekursif akan semakin tidak efisien ketika menghitung hampiran turunan fungsi dengan tingkatan turunan dan orde ketelitian yang semakin tinggi, karena jumlah data (titik-titik partisi) yang digunakan semakin banyak.

Untuk mengatasi permasalahan efisiensi komputasi di atas, diperlukan rumus eksplisit beda hingga sehingga koefisien-koefisiennya dapat ditentukan tanpa melewati proses rekursif. Rumus eksplisit ini telah diformulasi oleh [3] berdasarkan observasi numerik dan pembuktiannya untuk turunan pertama telah dijustifikasi secara matematis oleh [4]. Selanjutnya [5] juga telah membuktikan rumus eksplisit beda hingga untuk turunan kedua.

Rumus eksplisit beda hingga ini dapat diimplementasikan dalam sebuah rancangan aplikasi komputasi melalui pemrograman MATLAB. Hal ini akan mempermudah seseorang untuk menghitung turunan fungsi secara numerik, apapun fungsinya dan berapapun orde ketelitian yang diinginkan. Program komputasi dari rumus eksplisit beda hingga untuk turunan pertama telah dikembangkan oleh [6]. Pada makalah ini program komputasi tersebut dilanjutkan untuk turunan kedua.

## METODE

Dalam (Khan dan Ohba, 1999) telah dikembangkan rumus eksplisit dari koefisien-koefisien beda hingga yang diformulasi berdasarkan deret Taylor. Misalkan suatu fungsi $f(x)$ akan dicari turunan numeriknya. Terlebih dahulu partisi domain $f(x)$ 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

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

dimana $f_k=f(x_k)$ dan $f_0^{(m)}=f^{(m)}(x_0)$ dengan $m$ menyatakan tingkatan turunan, dan $\mathrm{O}(T^{n+1})$ merepresentasikan suku sisa (galat). Deret Taylor (2) dikatakan memiliki orde ketelitian $n+1$.

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

$$f_0'' \approx \frac{1}{T^2}\sum_k g_k f_k, \quad (3)$$

Dimana koefisien $g_k$ dan iterator $k$ didefinisikan berdasarkan orde ketelitian

**JURTEKSI (Jurnal Teknologi dan Sistem Informasi)** ISSN 2407-1811 (Print)
Vol. VI No. 1, Des 2019, hlm. 65 - 70 ISSN 2550-0201 (Online)
DOI: https://doi.org/10.33330/jurteksi.v6i1.433
Available online at http://jurnal.stmikroyal.ac.id/index.php/jurteksi

dan jenis beda hingga sebagai berikut:

1. Beda maju dengan orde ketelitian $N-1$, rumus eksplisit koefisiennya diberikan oleh

$$g_k \equiv g_k^F = \begin{cases} \frac{(-1)^k 2N!}{k(N-k)!\,k!} \sum_{m=1, m\neq k}^{N} \frac{1}{m}, & k = 1,2,\dots,N, \\ -\sum_{m=1}^{N} g_m, & k = 0. \end{cases} \quad (4)$$

2. Beda mundur dengan orde ketelitian $N-1$, rumus eksplisit koefisiennya diberikan oleh

$$g_k \equiv g_k^B = -g_{-k}^F, \quad k = -N, -N+1, \dots, -1, 0 \quad (5)$$

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

$$g_k \equiv g_k^C = \begin{cases} -2\sum_{m=1}^{N} g_m, & k = 0, \\ \frac{(-1)^{k+1} 2!(N!)^2}{k^2(N-k)!(N+k)!}, & k = \pm 1, \pm 2, \dots, \pm N. \end{cases} \quad (6)$$

## HASIL DAN PEMBAHASAN

### Diagram Alir Program


**Gambar 1. Diagram Alir Program**

### Pemrograman MATLAB

Script file pada pemrograman MATLAB ditulis dalam format m-file yang berisi kumpulan perintah yang dapat disimpan dan dijalankan secara berulang. File script tersebut disimpan dengan memberikan nama tanpa spasi dan memiliki ekstensi .m.

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

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

```
function [df0,g]=turunan2(f,x0,n,h,t)

%Program ini mencari turunan kedua
%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 df0.
%Koefisien-koefisien diberikan oleh %g.

if t==1 | t==2
   n=n+1;
   s=0;
   for k=1:n
      s=s+1/k;
   end

   for k=1:n
      g(k)=(-1)^(k)*2*factorial(n)...
         /(k*factorial(n-k)...
         *factorial(k))*(s-1/k);
   end

   g0=-sum(g);
if t==1
   xn=x0+n*h;
   x=[x0:h:xn]';
   y=0;
```

**JURTEKSI (Jurnal Teknologi dan Sistem Informasi)** ISSN 2407-1811 (Print)
Vol. VI No. 1, Des 2019, hlm. 65 - 70 ISSN 2550-0201 (Online)
DOI: https://doi.org/10.33330/jurteksi.v6i1.433
Available online at http://jurnal.stmikroyal.ac.id/index.php/jurteksi

```
   for k=1:n
      y=y+g(k)*f(x(k+1));
   end
   df0=(g0*f(x(1))+y)/h^2;
   g=[g0 g];
      end

if t==2
   g=-g;
   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^2;
   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(k+n+1)=(-1)^(k+1)...
     *factorial(2)*factorial(n)^2 ...
         /(k^2*factorial(n-k)...
         *factorial(n+k));
      else g(n+1)=0;
      end
   end
   g(n+1)=-2*sum(g(n+2:end));

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

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

```
function df=fungsiturunan2(f,a,b,n,h,t)

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

for k=1:N
   df(k)=turunan2(f,x(k),n,h,t);
end

plot(x,df,'bo-');
title(['Plot Turunan Kedua Fungsi']);
xlabel('x');
ylabel('d^2f/dx^2');
xlim([a b]);
```

**Contoh**

Agar lebih memperjelas penggunaan metode beda hingga untuk turuna kedua pada program MATLAB, perhatikan contoh berikut.

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

*Penyelesaian.*
Dari informasi di atas diperoleh:

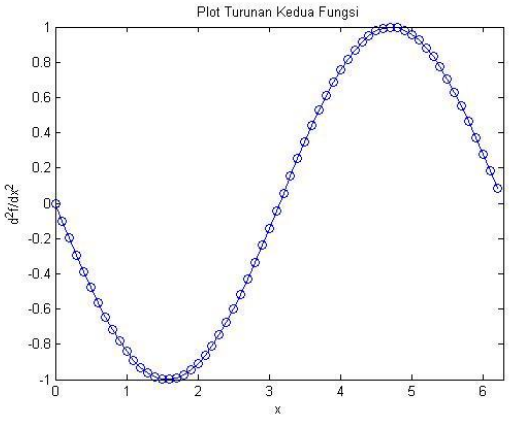
- $f(x) = \sin(x)$
- metode beda pusat ($t$=3)
- selang [a,b]=[0,2*pi]
- orde ketelitian 2 ($2n$=2 → $n$=1)
- lebar selang 0.1 ($h$=0.1)

Dengan menggunakan data di atas untuk variabel input, maka pemanggilan fungsi fungsiturunan2 dalam *command window* pada MATLAB dapat dilakukan dengan mengetik script berikut:

```
fungsiturunan2(@(x)sin(x),0,2*pi,2,0.1,3);
```

**JURTEKSI (Jurnal Teknologi dan Sistem Informasi)** ISSN 2407-1811 (Print)
Vol. VI No. 1, Des 2019, hlm. 65 - 70 ISSN 2550-0201 (Online)
DOI: https://doi.org/10.33330/jurteksi.v6i1.433
Available online at http://jurnal.stmikroyal.ac.id/index.php/jurteksi

Hasil plot fungsi turunan kedua diberikan pada Gambar 2 yang menunjukkan kurva sinus negatif. Hal ini sesuai dengan turunan kedua dari sin(*x*).




**Gambar 2. Grafik fungsi turunan kedua dari *f*(*x*) = sin(*x*)**

Untuk mengkonfirmasi kevalidan hasil yang diperoleh, perlu diperiksa galat dari turunan numerik di setiap orde ketelitian. Misalkan untuk fungsi pada contoh sebelumnya, akan dihitung galat dari turunan fungsi mulai dari orde ketelitian *nawal* sampai *nakhir*. Karena galat yang dihasilkan akan bernilai sangat kecil, maka untuk melihat perbedaan nilainya dengan lebih jelas, didefinisikan galat untuk setiap orde ketelitian 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 dengan *nawal*=2 dan *nakhir*=100. Karena yang digunakan adalah metode beda pusat, maka orde ketelitian yang digunakan haruslah bernilai genap. Dengan demikian langkah yang dipakai adalah sebesar 2.

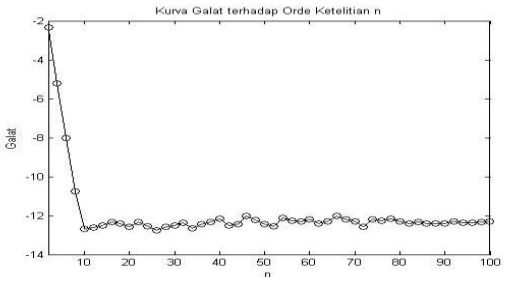
```
a=0;
b=2*pi;
h=0.1;
f=@(x)sin(x);
x=[a:h:b];
dfeksak=-sin(x);
t=3;

nawal=2;
nakhir=100;

k=1;
for n=nawal:2:nakhir
   dfnum=fungsiturunan2(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]);
```

Plot galat terhadap orde ketelitian *n* disajikan pada Gambar 3. Dari gambar dapat dilihat bahwa galat pada orde ketelitian 10 dan seterusnya tidak lagi semakin kecil sebagaimana diharapkan, namun berosilasi sampai orde ketelitian yang besar sekalipun. Hal ini disebabkan oleh kontribusi galat yang berasal dari pembulatan komputasi yang melebihi batas ketelitian pada MATLAB.




**Gambar 3. Galat turunan kedua dari *f*(*x*) = sin(*x*) terhadap orde ketelitian**

**JURTEKSI (Jurnal Teknologi dan Sistem Informasi)** ISSN 2407-1811 (Print)
Vol. VI No. 1, Des 2019, hlm. 65 - 70 ISSN 2550-0201 (Online)
DOI: https://doi.org/10.33330/jurteksi.v6i1.433
Available online at http://jurnal.stmikroyal.ac.id/index.php/jurteksi

## SIMPULAN

Pada makalah ini telah dibahas program aplikasi komputasi pada MATLAB untuk memperoleh turunan kedua suatu fungsi dengan menggunakan metode beda hingga untuk berbagai tipe (maju, mundur, atau pusat) dan sebarang orde ketelitian. Program tersebut dirancang dengan menggunakan rumus eksplisit metode beda hingga untuk turunan kedua yang diberikan dalam [3] dan dibuktikan dalam [5]. Melalui program MATLAB ini, seseorang dengan mudah dapat menghitung turunan kedua fungsi secara numerik dan visualisasinya dalam bentuk grafik, apapun fungsinya dan berapapun orde ketelitian yang diinginkan. Program MATLAB ini melanjutkan program serupa yang telah dibuat untuk turunan pertama dalam [6]. Program MATLAB ini dapat dikembangkan lebih lanjut untuk mendapatkan hampiran turunan fungsi dengan orde turunan dan orde ketelitian sebarang dengan menggunakan rumus eksplisit metode beda hingga diberikan dalam [3].

## 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. T/103/L1.3.1/PT.01.03/2019. Penulis juga mengucapkan terima kasih kepada STMIK/ AMIK Royal Kisaran dan Universitas Andalas atas dukungan dan fasilitas yang diberikan untuk penelitian ini.

## DAFTAR PUSTAKA

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

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

[3] 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.

[4] 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.

[5] Syafwan, H, Y. Y. Sutra, R. Alkhairi, M. Syafwan, W. Ramdhan dan R. A. Yusda. 2019. A Mathematical Proof of Explicit Formulas for the Coefficients of Finite Difference Approximations of Second Derivative. Malaysian Journal of Mathematical Sciences 13(3):359-371(2019).

[6] Syafwan, H, M. Syafwan, W. Ramdhan dan R. A. Yusda. 2018. Pemrograman Komputasi Rumus Eksplisit Metode Beda Hingga Untuk Turunan Pertama Dengan Menggunakan Matlab. Prosiding Seminar Nasional Royal (SENAR) 2018. Vol 1, No.1, Hal 61-66. ISSN 2622-6510.