UMEÅ UNIVERSITET
Department of Computer Science

January 13, 2022

5dv005ht21

Assignment 3 - Error estimation for artillery computations

Namn Mathias Hallberg
Gustaf Söderlund

Mail c19mhg@student.umu.se
et14gsd@student.umu.se

Version 1

Course
Scientific Computing
Handledare
Carl Christian Kjelgaard Mikkelsen Fredrik Peteri

Contents

1 Introduction **1**

2 MyRichardson **2**
2.1 Matlab Code . . . 3

3 a3range **5**
3.1 Determine the power of the primary error term and the secondary error term . . . 5
3.1.1 Method rk1 . . . 5
3.1.2 Method rk2 . . . 7
3.1.3 Method rk3 . . . 8
3.1.4 Method rk4 . . . 9
3.2 Behaviour of the computed value of Richardson's fraction . . . 10
3.3 Identifying the best approximation of the range . . . 10
3.4 Matlab Code for a3range.m . . . 10

4 a3time **12**
4.1 Matlab Code for a3time.m . . . 13

5 a3low **14**
5.1 Matlab Code for a3low.m . . . 17

6 a3range_g7 **19**
6.1 Matlab Code for a3range$_g$7 . . . 20

7 a3range_sabotage **22**
7.1 Matlab Code for a3range_sabotage . . . 25

8 a3length **27**
8.1 Method rk1 . . . 27
8.2 Method rk2 . . . 27
8.3 Method rk3 . . . 28
8.4 Method rk4 . . . 28
8.5 Matlab code for a3length . . . 29

9 Conclusion **30**

1 Introduction

This paper describes the process through the assignment. In the previous assignments we have faced the problems of approximating the range of a gun, flight time of a shell or the elevation that's needed to hit a given target. In this assignment we are challenged to apply the Richardson's techniques to compute reliable and accurate error estimations for these approximations.

In the assignment we look if there exists asymptotic error expansions on the form

$$T - A_h = \alpha h^p + \beta h^p + O(h^r), 0 < p < q < r \tag{1}$$

In Equation 1 we view A each approximation as a function of the size of the time step h used when computing the trajectories, that is $A = A_h$. The term αh^p is the primary error term, while βh^p is the secondary order term. The difference between the target value T and the approximation A_h is what the asymptotic error expansion describes.

Richardson's fraction is defined in Equation 2 and is frequently used to solve the tasks in this assignment.

$$F_h = \frac{A_{2h} - A_{4h}}{A_h - A_{2h}} \tag{2}$$

2 MyRichardson

When running the program MyRichardson.m together with the provided minimal working example a3f2. We get the following graph printed that's presented in Figure 1. We do also get a table with the data that was given that can be shown in the Figure

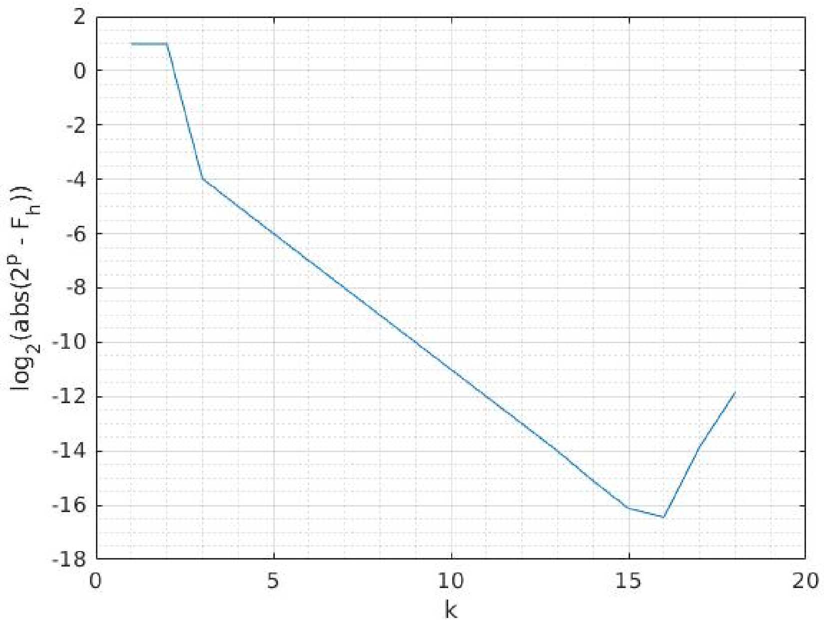


Figure 1: Result of plot by using a3f2_mwe

2.1 Matlab Code

```matlab
function data=MyRichardson(a,p,t)

% A3F1 Computational kernel for Richardson's technique
%
% Does Richardson extrapolation for a set of values assuming that the
% user has determined the order of the primary error term correctly
%
% CALL SEQUENCE: data=richardson(val,p);
%
% INPUT:
%    a        array of m approximations of t, such that if a(i) corresponds
%             stepsize h, then a(i+1) corresponds to stepsize h/2
%    p        the order of the primary order term
%    t        (optional) the target value of the approximations
%
% OUTPUT:
%    data    an array of information such that
%               data(i,1) = i
%               data(i,2) = a(i)
%               data(i,3) = Richardson's fraction for i > 2
%               data(i,4) = Richardson's error estimate for i > 1
%             if the exact target value is supplied, then
%               data(i,5) = exact error
%               data(i,6) = comparision of error estimate to exact error
%
% MINIMAL WORKING EXAMPLE: A3F2

% PROGRAMMING by Carl Christian K. Mikkelsen (spock@cs.umu.se)
%                Mathias Hallberg (c19mhg@cs.umu.se)
%                Gustaf Soderlund (et14gsd@cs.umu.se)
%   2015-12-10  Initial programming amd testing
%   2018-12-09  Printing moved to minimal working example
%   2018-12-09  Skeleton extracted from working code
%   2022-01-13  Finished the skeleton

% Reshape the input array as a colum vector
m=numel(a); a=reshape(a,m,1);

% Is the target value known?
if ~exist('target','var')
    % Set a flag to indicate that the target value is unknown
    flag=0;
```

```
    % Allocate space for the table used to print the results
    data=zeros(m,4);
else
    % Set a flag to indicate that the the target value is known
    flag=1;
    % Allocate space for the table used to print the results
    data=zeros(m,6);
end

% Initialize the first and the second columns of data
for i=1:m
   data(i,1) = i;
   data(i,2) = a(i);

end

% Process the data, computing Richardson's fractions
for i=3:m

    F_h = (a(i-1)-a(i-2))/(a(i)-a(i-1));
    data(i,3) = F_h;
end

% Compute Richardson's error estimates assuming order p is correct!
for i=2:m
   E_h = (a(i)-a(i-1))/(2^p-1);
   data(i,4) = E_h;
end

% If possible, then compute the error and compare it to the error estimate
if (flag==1)
    for i=1:m
        % Compute the exact error

        % Compare the error estimate to the true error
        % i.e. log10(abs(relative error))

    end
end
```

3 a3range

In this section we will use the script a3range to calculate the range of the artillery and estimate it using Richardson's fraction. With the estimation we will analyze the approximation and the error that follows with it.

3.1 Determine the power of the primary error term and the secondary error term

By using the Equation 2 we can redefine it as

$$F_h - 2^p = O(h^m), h \to 0, h > 0 \tag{3}$$

To determine the power of the primary error term we can then use Equation 4.

$$F_h = 2^p + O(h^m), h \to 0, h > 0 \tag{4}$$

Because h converges towards zero we can get the power of the primary order term via Equation 5.

$$log_2(p) = log_2(F_h) \tag{5}$$

The secondary order term is the absolute value of the derivative or the slope of the graphs on arbitrary values. The result of the derivative will result in being $|m|$ which can be combined using $m = q - p$ and therefore q can be solved by $q = m + p$

3.1.1 Method rk1

To find the power of the primary error term we can use the Equations 4 and 5 together with the data provided from the graph in Figure 2 and the table from Figure 3.

By inspecting the table in Figure 3 we can see that the Richardson's fraction is moving towards 2, i.e $F_h \approx 2$. By using Equation 5 we can decide the power of the primary error term is $log_2(2) = 1$.

To get the power of the secondary error term we calculate the following way. By inspecting the graph in Figure 2 we can see that the slope is -1 but using the absolute value of -1 gives us 1 from the interval $[3, 5]$. If the slope $|m| = 1$ then we get the value $q = 2$ by using $q = m + p$. In other words we just calculated the power of the secondary order term that's 2.

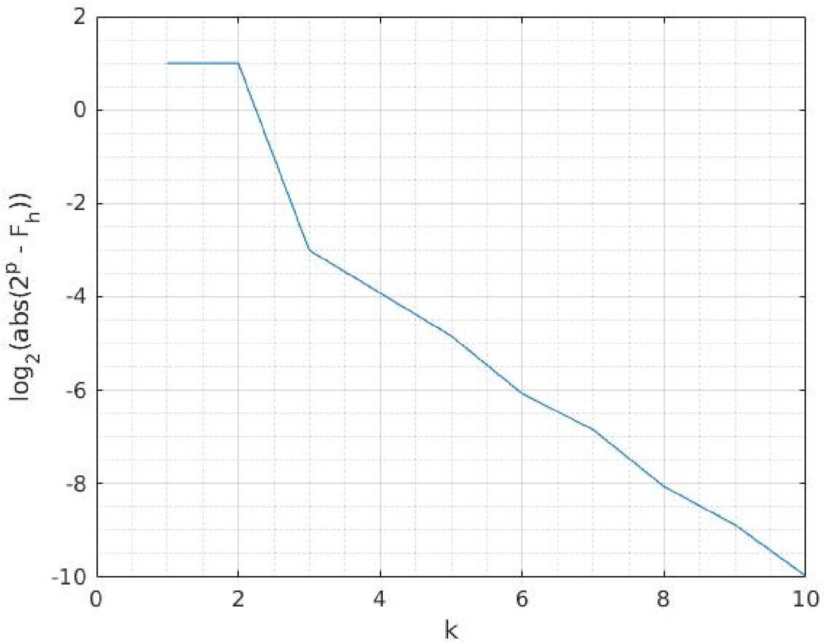


Figure 2: Result of plot of method rk1.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.215475800698e+04	0.00000000	0.000000000000e+00
2	2.226834769726e+04	0.00000000	1.135896902761e+02
3	2.232180737277e+04	2.12477328	5.345967551322e+01
4	2.234768231668e+04	2.06607890	2.587494390464e+01
5	2.236039746277e+04	2.03497024	1.271514609780e+01
6	2.236670811603e+04	2.01487003	6.310653262091e+00
7	2.236984977888e+04	2.00869844	3.141662846803e+00
8	2.237141767501e+04	2.00374425	1.567896123950e+00
9	2.237220079351e+04	2.00211861	7.831185015530e-01
10	2.237259215891e+04	2.00099062	3.913654031312e-01

Figure 3: Result of table of method rk1.

3.1.2 Method rk2

To find the power of the primary error term we can use the Equations 4 and 5 together with the data provided from the graph in Figure 4 and the table from Figure 5.

By inspecting the table in Figure 5 we can see that the Richardson's fraction is moving towards 4, i.e $F_h \approx 4$. By using Equation 5 we can decide the power of the primary error term is $log_2(4) = 2$.

To get the power of the secondary error term we calculate the following way. By inspecting the graph in Figure 4 we can see that the slope is -1 but using the absolute value of -1 gives us 1 from the interval $[3, 10]$. If the slope $|m| = 1$ then we get the value $q = 3$ by using $q = m + p$. In other words we just calculated the power of the secondary order term that's 3.

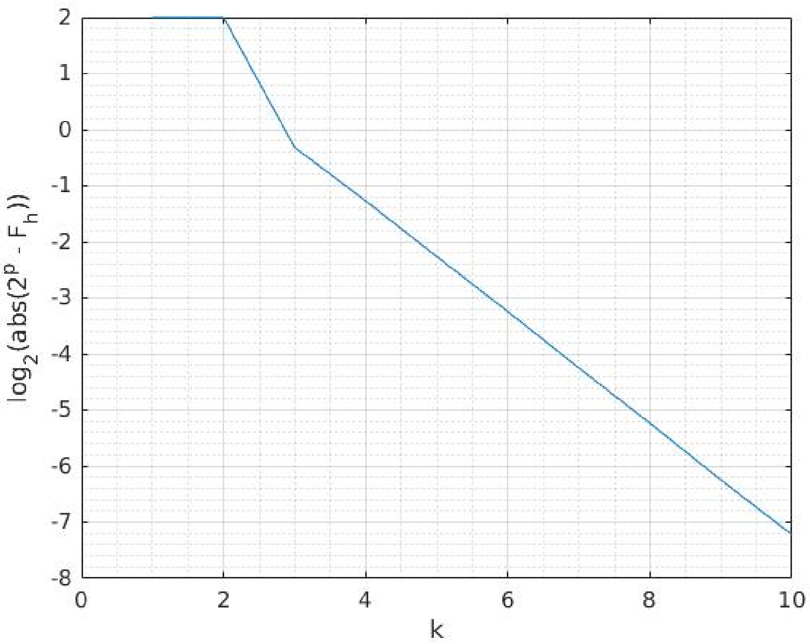


Figure 4: Result of plot of method rk2.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.236937113617e+04	0.00000000	0.000000000000e+00
2	2.237221489000e+04	0.00000000	9.479179419698e-01
3	2.237280705137e+04	4.80232914	1.973871250727e-01
4	2.237294120066e+04	4.41419684	4.471643019982e-02
5	2.237297307937e+04	4.20811609	1.062623493393e-02
6	2.237298084435e+04	4.10544520	2.588327064586e-03
7	2.237298276044e+04	4.05251784	6.386960327897e-04
8	2.237298323630e+04	4.02651976	1.586223515915e-04
9	2.237298335488e+04	4.01313863	3.952575934818e-05
10	2.237298338448e+04	4.00668004	9.864965250017e-06

Figure 5: Result of table of method rk2.

3.1.3 Method rk3

To find the power of the primary error term we can use the Equations 4 and 5 together with the data provided from the graph in Figure 6 and the table from Figure 7.

By inspecting the table in Figure 7 we can see that the Richardson's fraction is moving towards 8, i.e $F_h \approx 8$. By using Equation 5 we can decide the power of the primary error term is $log_2(8) = 3$.

To get the power of the secondary error term we calculate the following way. By inspecting the graph in Figure 6 we can see that the slope is -1 but using the absolute value of -1 gives us 1 from the interval $[3, 8]$. If the slope $|m| = 1$ then we get the value $q = 4$. In other words we just calculated the power of the secondary order term that's 4.

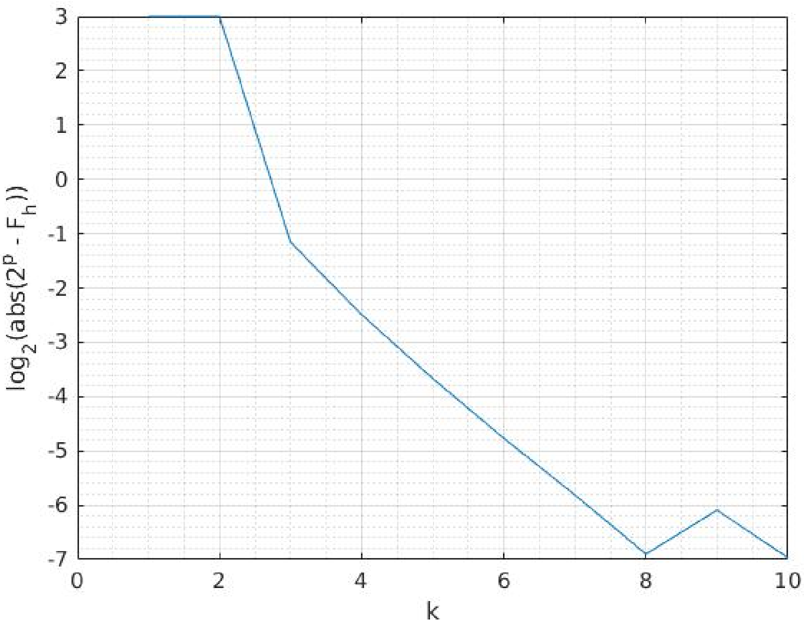


Figure 6: Result of plot of method rk3.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.237327039413e+04	0.00000000	0.000000000000e+00
2	2.237302121565e+04	0.00000000	-3.559692690968e-02
3	2.237298822139e+04	7.55217806	-4.713464993464e-03
4	2.237298400328e+04	7.82203872	-6.025877862287e-04
5	2.237298347078e+04	7.92134046	-7.607144133155e-05
6	2.237298340391e+04	7.96329830	-9.552755467926e-06
7	2.237298339553e+04	7.98220289	-1.196756784339e-06
8	2.237298339448e+04	7.99163610	-1.497511610588e-07
9	2.237298339435e+04	7.98533976	-1.875326103930e-08
10	2.237298339433e+04	7.99202658	-2.346496330574e-09

Figure 7: Result of table of method rk3.

3.1.4 Method rk4

To find the power of the primary error term we can use the Equations 4 and 5 together with the data provided from the graph in Figure 8 and the table from Figure 9.

By inspecting the table in Figure 9 we can see that the Richardson's fraction is moving towards 16, i.e $F_h \approx 16$. By using Equation 5 we can decide the power of the primary error term is $log_2(16) = 4$.

To get the power of the secondary error term we calculate the following way. By inspecting the graph in Figure 8 we can see that the slope is -1 but using the absolute value of -1 gives us 1 from the interval $[3, 6]$. If the slope $|m| = 1$ then we get the value $q = 5$. In other words we just calculated the power of the secondary order term that's 5.

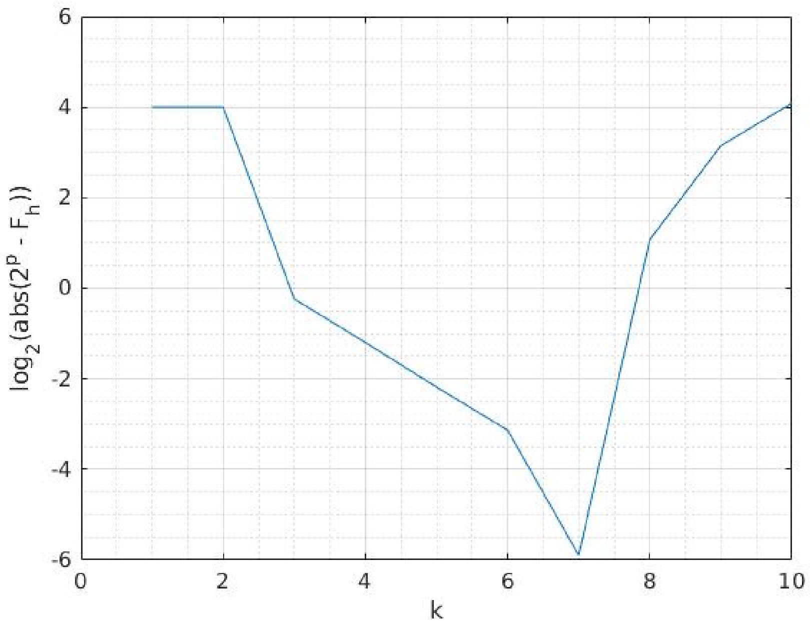


Figure 8: Result of plot of method rk4.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.237296894265e+04	0.00000000	0.000000000000e+00
2	2.237298253527e+04	0.00000000	9.061748710034e-04
3	2.237298334202e+04	16.84861431	5.378334705407e-05
4	2.237298339111e+04	16.43565957	3.272357086341e-06
5	2.237298339413e+04	16.22025681	2.017450848750e-07
6	2.237298339432e+04	16.11384681	1.251998279865e-08
7	2.237298339433e+04	16.01675458	7.816803796838e-10
8	2.237298339433e+04	18.10674157	4.317068184415e-11
9	2.237298339433e+04	7.12000000	6.063298011820e-12
10	2.237298339433e+04	-1.04166667	-5.820766091347e-12

Figure 9: Result of table of method rk4.

3.2 Behaviour of the computed value of Richardson's fraction

By inspecting the behaviour of the values from the calculations we can determine that the computed values of Richardson's fraction behaves like a asymptotic error expansion between the intervals for each method. Method rk1 at the interval [3,5], method rk2 at the interval [3,10], method rk3 at the interval [3,8] and method rk4 at the interval [3,6].

3.3 Identifying the best approximation of the range

To identify the best approximation of the range we inspected the data given by the program and decided that $k = 6$ in Figure 9 is the most accurate one due to the low E_h, error estimate.

3.4 Matlab Code for a3range.m

```
% Script which uses MyRichardson to get an approximation
    of range from range_rkx
%
% PROGRAMMING by  Mathias Hallberg (c19mhg@cs.umu.se)
%                 Gustaf Soderlund (et14gsd@cs.umu.se)
%
%   2022-01-13  Finished the program


% Clean up
clear all

% Load parameters describing shot
a3f3

% Set initial time step
h0=1;

% Methods
m=["rk1","rk2","rk3","rk4"];

% Number of rows in table
kmax=10;

% Loop over methods
for i=1:4
    % Select method
    method=m(i);

    % Initialize time step
    dt=h0;

    % Initialize maxstep
    maxstep=200;

```

```
    % Loop over approximations
    for k=1:kmax
      % Compute range
      [r, flag, t, tra]=range_rkx(param,v0,theta,method,
          dt,maxstep);
      % Save information
      a(k)=r;
      % Decrease time step
      dt=2^-k;
      % Increase maxstep
      maxstep=maxstep*2;
    end

    % Run Richardsons techniques
    data=MyRichardson(a,i);

    % New figure
    h(i)=figure();

    % Print to screen
    rdifprint(data,i);
end
```

4 a3time

In the program a3time one method were to be chosen, the method that was chosen was rk1. The result of the program is given by the Figure 10 and Figure 11. By inspecting the data we can say that the error estimate can be trusted due to having an asymptotic error expansion when using Richardson's fraction. By inspecting the interval $[3, 5]$ in Figure 10 we can see that the graph is linear, thus providing us with a solution for the primary error term and secondary error term.

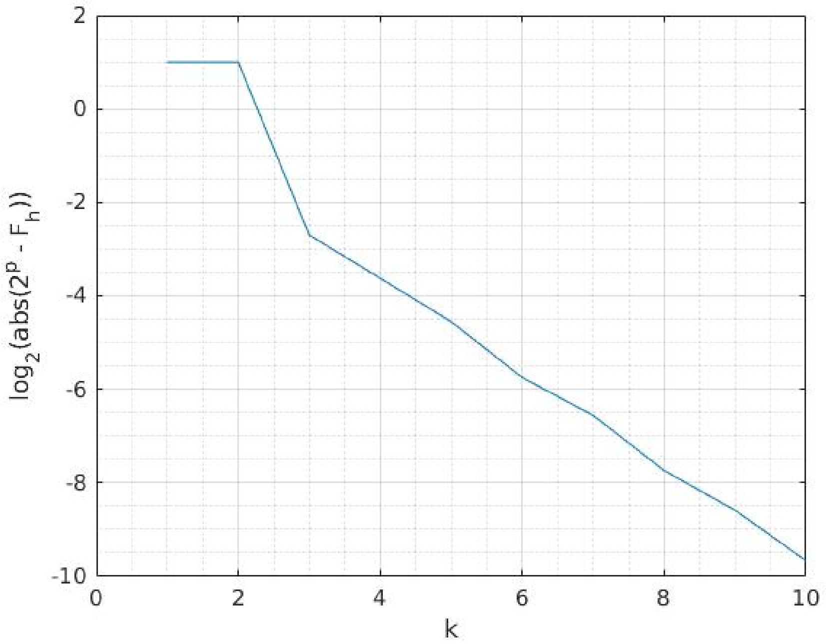


Figure 10: Result of a3time of method rk1

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	7.806167346462e+01	0.00000000	0.000000000000e+00
2	7.824770410760e+01	0.00000000	1.860306429748e-01
3	7.833406544790e+01	2.15409629	8.636134030678e-02
4	7.837556298903e+01	2.08111946	4.149754112385e-02
5	7.839587909653e+01	2.04259311	2.031610750275e-02
6	7.840594352718e+01	2.01860475	1.006443065204e-02
7	7.841094918058e+01	2.01061277	5.005653401085e-03
8	7.841344616212e+01	2.00468178	2.496981540162e-03
9	7.841469303448e+01	2.00259596	1.246872355409e-03
10	7.841531608840e+01	2.00122704	6.230539204921e-04

Figure 11: Result of table of method rk4.

4.1 Matlab Code for a3time.m

```
% Script which uses MyRichardson to get an approximation
  of time from range_rkx
%
% PROGRAMMING by  Mathias Hallberg (c19mhg@cs.umu.se)
%                 Gustaf Soderlund (et14gsd@cs.umu.se)
%
%   2022-01-13  Finished the program
% Clean up

clear all

% Load parameters describing shot
a3f3

% Set initial time step
h0=1;

% Number of rows in table
kmax=10;

% Initialize time step
dt=h0;

% Select method
method = 'rk1';
% Initialize maxstep
maxstep=200;

% Loop over approximations
for k=1:kmax
  % Compute range
  [r, flag, t, tra]=range_rkx(param,v0,theta,method,dt,
      maxstep);
  % Save information
  a(k)=t(:, end);
  % Decrease time step
  dt=2^-k;
  % Increase maxstep
  maxstep=maxstep*2;
end

% Run Richardsons techniques
data=MyRichardson(a, 1);

% New figure
h(1)=figure();

% Print to screen
rdifprint(data, 1);
```

5 a3low

The script a3low will compute a low firing solution for a projectile that will hit a target located at 15 000 meters to the right of the gun. Since Richardson's fraction will not behave correctly unless the elevations are computed with excessive accuracy. The residual is set to as small as 10^{-10} meters.

The result of the program is shown as graphs in Figure 12, 13, 14, 15 and as table in Figure 16, 17, 18, 19.

We can see in the result of the program shown in the figures that F_h is moving towards the primary error term for each method. The values are somewhat linear in specific intervals.

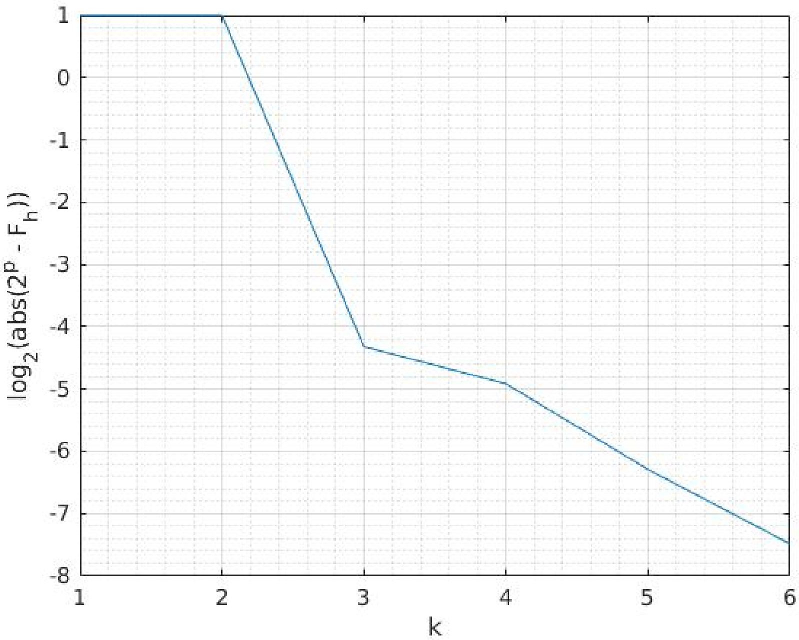


Figure 12: Result of graph using method rk1 on a low firing shot.

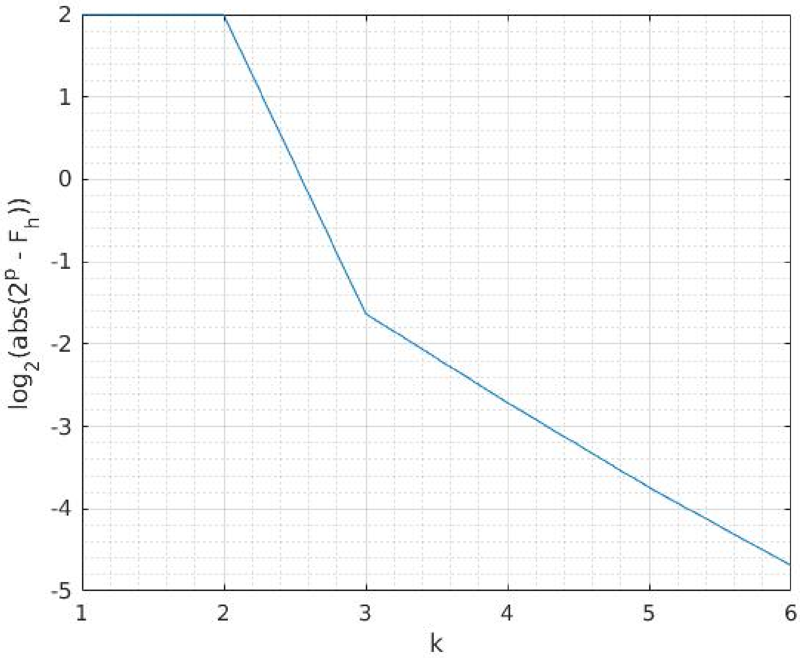


Figure 13: Result of graph using method rk2 on a low firing shot.

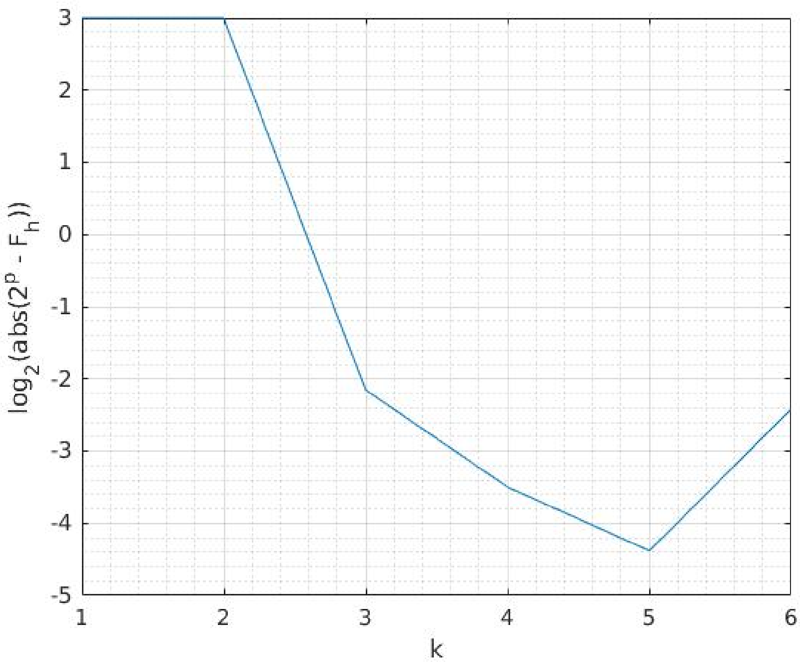


Figure 14: Result of graph using method rk3 on a low firing shot.

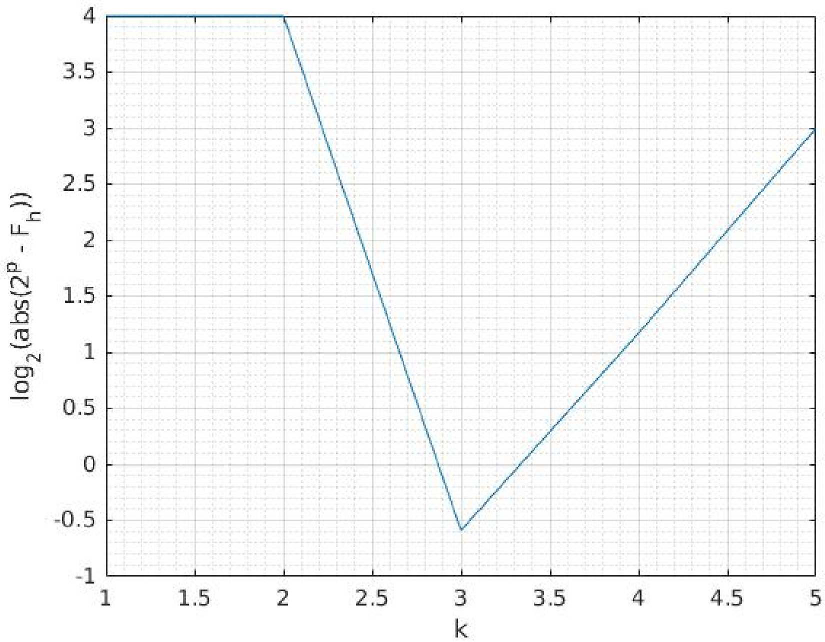


Figure 15: Result of graph using method rk4 on a low firing shot.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.697098682032e-01	0.00000000	0.000000000000e+00
2	2.723615949903e-01	0.00000000	2.651726787133e-03
3	2.737215137124e-01	1.94991564	1.359918722050e-03
4	2.744129243748e-01	1.96687554	6.914106624294e-04
5	2.747608482388e-01	1.98724702	3.479238640369e-04
6	2.749352961405e-01	1.99442848	1.744479017010e-04

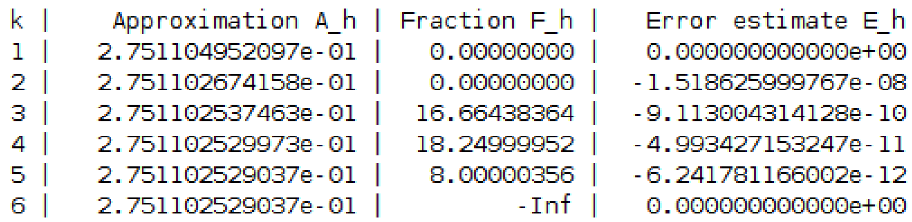
Figure 16: Result of table of method rk1 on a low firing shot.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.752864069658e-01	0.00000000	0.000000000000e+00
2	2.751514417484e-01	0.00000000	-4.498840578769e-05
3	2.751202235915e-01	4.32329231	-1.040605228812e-05
4	2.751127061120e-01	4.15274249	-2.505826526333e-06
5	2.751108611967e-01	4.07470185	-6.149717493390e-07
6	2.751104043918e-01	4.03873745	-1.522683159561e-07

Figure 17: Result of table of method rk2 on a low firing shot.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.751036855484e-01	0.00000000	0.000000000000e+00
2	2.751094101692e-01	0.00000000	8.178029725240e-07
3	2.751101464500e-01	7.77505086	1.051829739577e-07
4	2.751102395150e-01	7.91146882	1.329499949814e-08
5	2.751102512184e-01	7.95199996	1.671906383180e-09
6	2.751102527164e-01	7.81249986	2.140040208535e-10

Figure 18: Result of table of method rk3 on a low firing shot.



k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.751104952097e-01	0.00000000	0.000000000000e+00
2	2.751102674158e-01	0.00000000	-1.518625999767e-08
3	2.751102537463e-01	16.66438364	-9.113004314128e-10
4	2.751102529973e-01	18.24999952	-4.993427153247e-11
5	2.751102529037e-01	8.00000356	-6.241781166002e-12
6	2.751102529037e-01	-Inf	0.000000000000e+00

Figure 19: Result of table of method rk4 on a low firing shot.

5.1 Matlab Code for a3low.m

```
% Script which uses MyRichardson to get an approximation of of the
% angle to shoot to reach 15000 m
%
%
% PROGRAMMING by  Mathias Hallberg (c19mhg@cs.umu.se)
%                 Gustaf Soderlund (et14gsd@cs.umu.se)
%
%  2022-01-13  Finished the program

% Clean up
clear all

% Load parameters describing shot
a3f3

% Set initial time step
h0=1;

% Methods
m=["rk1","rk2","rk3","rk4"];

% Number of rows in table
kmax=6;

eps=0;delta=10^-10;
deg=linspace(0,45,1001);
rad=deg.*pi/180;
% Loop over methods
for i=1:4
    % Select method
    method=m(i);
    % Initialize time step
    dt=h0;
    % Initialize maxstep
    maxstep=200;
    % Loop over approximations
    for k=1:kmax
        table=compute_range(param,v0,rad,method,dt,
            maxstep, 'false');
        for j=1:numel(table(2,:))
            if (table(2,j)>15000)
```

```
                    thetapos=j;
                    break
                end
            end
        % Compute range
        r0=15000-range_rkx(param,v0,table(1,thetapos-1),
            method,dt,maxstep);
        r1=15000-range_rkx(param,v0,table(1,thetapos),
            method,dt,maxstep);
        % Save information
        thetafunc=@(theta)15000-range_rkx(param,v0,theta,
            method,dt,maxstep);
        [x,~,~,~,~,~,~]=bisection(thetafunc,table(1,
            thetapos-1),table(1,thetapos),r0,r1,delta,eps
            ,10001,'true');
        a(k)=x;
        % Decrease time step
        dt=2^-k;
        % Increase maxstep
        maxstep=maxstep*2;
    end
    % Run Richardsons techniques
    data=MyRichardson(a,i);
    % New figure
    h(i)=figure();
    % Print to screen
    rdifprint(data,i);
end
```

6 a3range_g7

The script a3range_g7 computes the range of the shot defined by a3a4. By inspecting the data given from the program shown in the Figures 20, 21, 22 and 23 it's clear that the value of F_h is not consistent moving towards primary error term. In the method rk1 we have a very large error estimate that can't be counted on. The method rk2 and rk3 give us a better accuracy. In method rk4 the F_h behaves very weird.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	1.664930524885e+04	0.00000000	0.000000000000e+00
2	1.676195575964e+04	0.00000000	1.126505107859e+02
3	1.681442430731e+04	2.14701027	5.246854767286e+01
4	1.683967765974e+04	2.07768643	2.525335242881e+01
5	1.685209336970e+04	2.03398376	1.241570995797e+01
6	1.685824620752e+04	2.01788350	6.152837827525e+00
7	1.686130901063e+04	2.00889108	3.062803102806e+00
8	1.686283702539e+04	2.00443293	1.528014761083e+00
9	1.686360018201e+04	2.00222958	7.631566214695e-01
10	1.686398154180e+04	2.00114602	3.813597885855e-01

Figure 20: Result of table using method rk1.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	1.686727894001e+04	0.00000000	0.000000000000e+00
2	1.686560270790e+04	0.00000000	-5.587440370012e-01
3	1.686469350754e+04	1.84363336	-3.030667861725e-01
4	1.686444431363e+04	3.64856573	-8.306463648357e-02
5	1.686438310476e+04	4.07120538	-2.040295902649e-02
6	1.686436783474e+04	4.00843556	-5.090005499369e-03
7	1.686436403115e+04	4.01462920	-1.267864413724e-03
8	1.686436308275e+04	4.01053407	-3.161335601665e-04
9	1.686436284598e+04	4.00567731	-7.892137485517e-05
10	1.686436278683e+04	4.00288654	-1.971611588184e-05

Figure 21: Result of table using method rk2.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	1.686490292933e+04	0.00000000	0.000000000000e+00
2	1.686438343416e+04	0.00000000	-7.421359496532e-02
3	1.686436019580e+04	22.35506532	-3.319766411421e-03
4	1.686436251379e+04	-10.02521996	3.311415036608e-04
5	1.686436271626e+04	11.44850072	2.892444276118e-05
6	1.686436275939e+04	4.69421763	6.161717465147e-06
7	1.686436276611e+04	6.42524393	9.589857654646e-07
8	1.686436276700e+04	7.54210180	1.271509972867e-07
9	1.686436276711e+04	7.86678457	1.616302012865e-08
10	1.686436276712e+04	7.84561049	2.060135427330e-09

Figure 22: Result of table using method rk3.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	1.686466800168e+04	0.00000000	0.000000000000e+00
2	1.686438149441e+04	0.00000000	-1.910048505936e-02
3	1.686436212153e+04	14.78909254	-1.291525156800e-03
4	1.686436290079e+04	-24.86056730	5.195075160979e-05
5	1.686436277444e+04	-6.16735346	-8.423508067305e-06
6	1.686436276728e+04	17.64433189	-4.774058955566e-07
7	1.686436276713e+04	48.01036585	-9.943808739384e-09
8	1.686436276713e+04	46.64391354	-2.131855580956e-10
9	1.686436276713e+04	4.65079365	-4.583853296936e-11
10	1.686436276713e+04	-94.50000000	4.850638409456e-13

Figure 23: Result of table using method rk4.

6.1 Matlab Code for a3range$_g$7

```
% Script which uses MyRichardson to get an approximation of range from range_rkx
% and using the parameters of the shot from a3f4
%
% PROGRAMMING by  Mathias Hallberg (c19mhg@cs.umu.se)
%                 Gustaf Soderlund (et14gsd@cs.umu.se)
%
%   2022-01-13  Finished the program


% Clean up
clear all

% Load parameters describing shot
a3f4

% Set initial time step
h0=1;

% Methods
m=["rk1","rk2","rk3","rk4"];

% Number of rows in table
kmax=10;

% Loop over methods
for i=1:4
    % Select method
    method=m(i);

    % Initialize time step
    dt=h0;

    % Initialize maxstep
    maxstep=200;

    % Loop over approximations
    for k=1:kmax
```

```
        % Compute range
        [r, flag, t, tra]=range_rkx(param,v0,theta,method,
            dt,maxstep);
        % Save information
        a(k)=r;
        % Decrease time step
        dt=2^-k;
        % Increase maxstep
        maxstep=maxstep*2;
    end

    % Run Richardsons techniques
    data=MyRichardson(a,i);

    % New figure
    h(i)=figure();

    % Print to screen
    rdifprint(data,i);
end
```

7 a3range_sabotage

The script a3range_sabotage uses the script range_rkx_sabotage to compute the range of the shot given by the script a3f3. In range_rkx_sabotage the tolerance is much larger then in range_rkx. This will gives the output data a problem with the accuracy.

By inspecting the graphs and tables given from the program shown in the Figures 24 25 26 27, 28, 29, 30 and 31 we can see that there's a pattern in the graphs. The graph spikes more then in the previous programs.
Due to the much larger tolerance in the range_rkx_sabotage none of the methods retain the ability to estimate the range accurately.

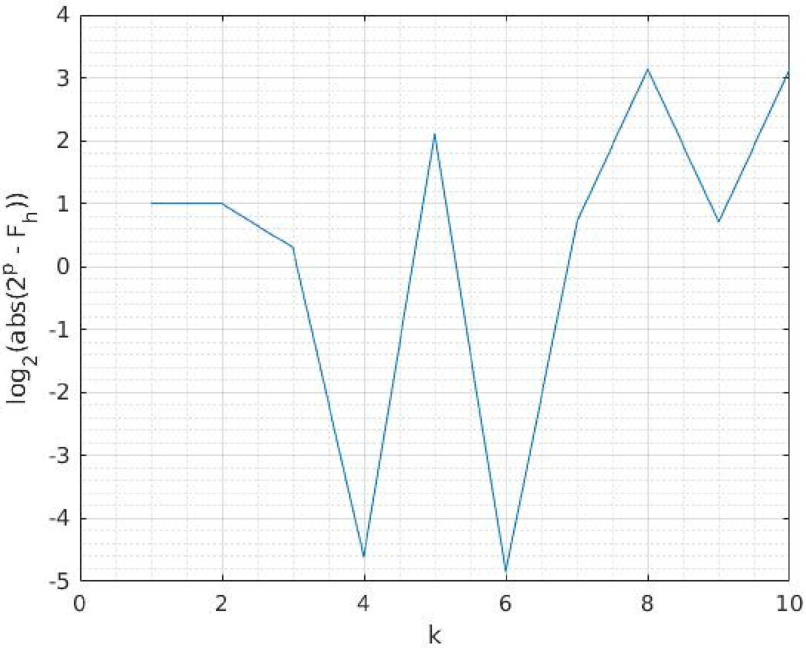


Figure 24: Result of graph using method rk1 using range_rkx_sabotage.

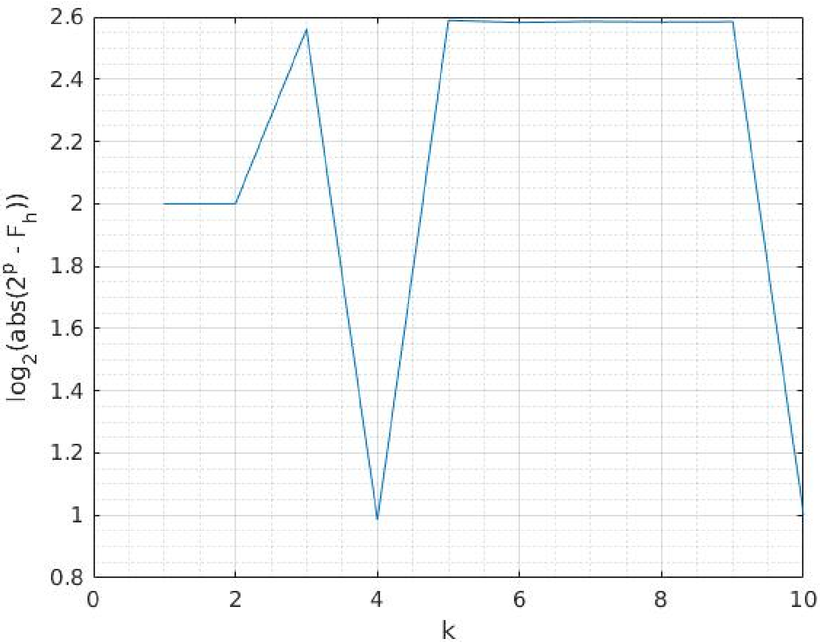


Figure 25: Result of graph using method rk2 using range_rkx_sabotage.

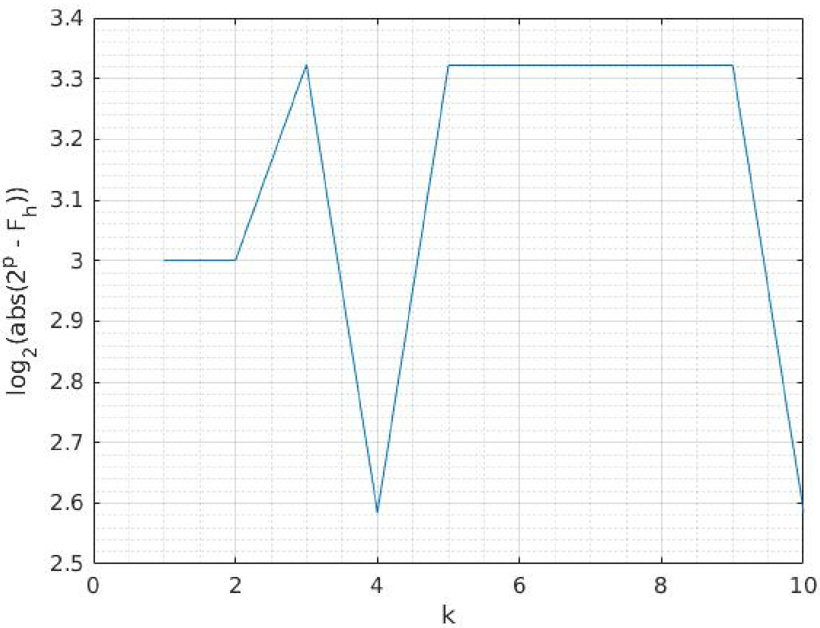


Figure 26: Result of graph using method rk3 using range_rkx_sabotage.

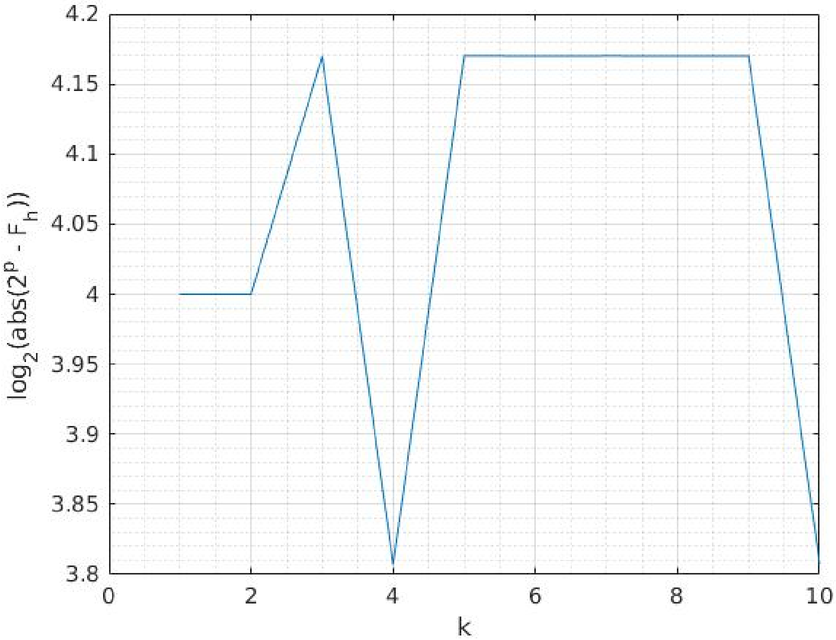


Figure 27: Result of graph using method rk4 using range_rkx_sabotage.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.222319538948e+04	0.00000000	0.000000000000e+00
2	2.226870973610e+04	0.00000000	4.551434662380e+01
3	2.232825442554e+04	0.76437289	5.954468943185e+01
4	2.235743066698e+04	2.04086224	2.917624144809e+01
5	2.236203162744e+04	6.34133715	4.600960456646e+00
6	2.236429296702e+04	2.03461723	2.261339575569e+00
7	2.237034040403e+04	0.37393355	6.047437012112e+00
8	2.237089914713e+04	10.82328721	5.587430967389e-01
9	2.237240922481e+04	0.37000951	1.510077683051e+00
10	2.237254849996e+04	10.84240580	1.392751489147e-01

Figure 28: Result of table of method rk1 using range_rkx_sabotage.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.238441612822e+04	0.00000000	0.000000000000e+00
2	2.234642041878e+04	0.00000000	-1.266523648098e+01
3	2.236644902748e+04	-1.89707183	6.676202900556e+00
4	2.237636192965e+04	2.02045863	3.304300725150e+00
5	2.237145179067e+04	-2.01886387	-1.636712995657e+00
6	2.237391774200e+04	-1.99117433	8.219837771709e-01
7	2.237268740123e+04	-2.00428319	-4.101135912485e-01
8	2.237330321973e+04	-1.99789509	2.052728356321e-01
9	2.237299547119e+04	-2.00104443	-1.025828475067e-01
10	2.237284157875e+04	1.99976383	-5.129748121544e-02

Figure 29: Result of table of method rk2 using range_rkx_sabotage.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.238640817662e+04	0.00000000	0.000000000000e+00
2	2.234680510099e+04	0.00000000	-5.657582232883e+00
3	2.236653158750e+04	-2.00760919	2.818069500944e+00
4	2.237638089793e+04	2.00282920	1.407044348083e+00
5	2.237145632573e+04	-2.00003372	-7.035103140765e-01
6	2.237391885002e+04	-1.99980655	3.517891840150e-01
7	2.237268767502e+04	-2.00014156	-1.758821427434e-01
8	2.237330328778e+04	-1.99991794	8.794467967891e-02
9	2.237299548815e+04	-2.00004386	-4.397137560383e-02
10	2.237284158298e+04	1.99993039	-2.198645304712e-02

Figure 30: Result of table of method rk3 using range_rkx_sabotage.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.238620789796e+04	0.00000000	0.000000000000e+00
2	2.234677942387e+04	0.00000000	-2.628564939916e+00
3	2.236652834193e+04	-1.99648781	1.316594537693e+00
4	2.237638049035e+04	2.00452909	6.568098948078e-01
5	2.237145627471e+04	-2.00075487	-3.282810426419e-01
6	2.237391884364e+04	-1.99962551	1.641712616799e-01
7	2.237268767423e+04	-2.00018689	-8.207796089797e-02
8	2.237330328768e+04	-1.99990660	4.104089707107e-02
9	2.237299548814e+04	-2.00004669	-2.051996944623e-02
10	2.237284158298e+04	1.99992996	-1.026034403355e-02

Figure 31: Result of table of method rk4 using range_rkx_sabotage.

7.1 Matlab Code for a3range_sabotage

```
% Script which uses MyRichardson to get an approximation
    of range from range_rkx_sabotage
%
% PROGRAMMING by   Mathias Hallberg (c19mhg@cs.umu.se)
%                  Gustaf Soderlund (et14gsd@cs.umu.se)
%
%   2022-01-13   Finished the program

% Clean up
clear all

% Load parameters describing shot
a3f3

% Set initial time step
h0=1;

% Methods
m=["rk1","rk2","rk3","rk4"];

% Number of rows in table
kmax=10;

% Loop over methods
for i=1:4
```

```
    % Select method
    method=m(i);

    % Initialize time step
    dt=h0;

    % Initialize maxstep
    maxstep=200;

    % Loop over approximations
    for k=1:kmax
      % Compute range
      [r, flag, t, tra]=range_rkx_sabotage(param,v0,theta
          ,method,dt,maxstep);
      % Save information
      a(k)=r;
      % Decrease time step
      dt=2^-k;
      % Increase maxstep
      maxstep=maxstep*2;
    end

    % Run Richardsons techniques
    data=MyRichardson(a,i);

    % New figure
    h(i)=figure();

    % Print to screen
    rdifprint(data,i);
end
```

8 a3length

In the script a3length we will compute the length of the trajectory while using the trapezoidal rule. For each method rk1, rk2, rk3 and rk4 we are finding the primary error term.

8.1 Method rk1

By inspecting the table in Figure 32 we can see that the Richardson's fraction is moving towards 2, i.e $F_h \approx 2$. By using Equation 5 we can decide the power of the primary error term to $log_2(2) = 1$.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.778550749350e+04	0.00000000	0.000000000000e+00
2	2.797266970875e+04	0.00000000	1.871622152490e+02
3	2.806264768086e+04	2.08008928	8.997797210782e+01
4	2.810674546756e+04	2.04041923	4.409778670096e+01
5	2.812856952715e+04	2.02060421	2.182405959311e+01
6	2.813943058101e+04	2.00938692	1.086105385865e+01
7	2.814484731193e+04	2.00509385	5.416730916528e+00
8	2.814755250469e+04	2.00234563	2.705192760979e+00
9	2.814890425452e+04	2.00125252	1.351749831345e+00
10	2.814957992551e+04	2.00060362	6.756709924593e-01

Figure 32: Result of table of method rk1 using trapezoidal rule.

8.2 Method rk2

By inspecting the table in Figure 33 we can see that the Richardson's fraction is moving towards 2, i.e $F_h \approx 4$. By using Equation 5 we can decide the power of the primary error term to $log_2(4) = 2$.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.815272680908e+04	0.00000000	0.000000000000e+00
2	2.815101077100e+04	0.00000000	-5.720126932235e-01
3	2.815046025197e+04	3.11712766	-1.835063418839e-01
4	2.815030851983e+04	3.62822946	-5.057738046526e-02
5	2.815026895635e+04	3.83515635	-1.318782752181e-02
6	2.815025886221e+04	3.91945055	-3.364713332606e-03
7	2.815025631423e+04	3.96162851	-8.493258076972e-04
8	2.815025567411e+04	3.98047048	-2.133732208070e-04
9	2.815025551370e+04	3.99065106	-5.346827310859e-05
10	2.815025547355e+04	3.99508585	-1.338351042553e-05

Figure 33: Result of table of method rk2 using trapezoidal rule.

8.3 Method rk3

By inspecting the table in Figure 34 Richardson's fraction is moving towards 2, i.e $F_h \approx 4$. By using Equation 5 we can decide the power of the primary error term to $log_2(4) = 2$.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.815469282472e+04	0.00000000	0.000000000000e+00
2	2.815133289885e+04	0.00000000	-4.799894097965e-01
3	2.815052033276e+04	4.13495703	-1.160808700811e-01
4	2.815032107762e+04	4.07801817	-2.846502032064e-02
5	2.815027178877e+04	4.04260035	-7.041264998926e-03
6	2.815025953261e+04	4.02155947	-1.750879242796e-03
7	2.815025647707e+04	4.01112984	-4.365052523748e-04
8	2.815025571424e+04	4.00548737	-1.089768141225e-04
9	2.815025552366e+04	4.00282635	-2.722496667827e-05
10	2.815025547603e+04	4.00134657	-6.803951172125e-06

Figure 34: Result of table of method rk3 using trapezoidal rule.

8.4 Method rk4

By inspecting the table in Figure 35 Richardson's fraction is moving towards 2, i.e $F_h \approx 4$. By using Equation 5 we can decide the power of the primary error term to $log_2(4) = 2$.

k	Approximation A_h	Fraction F_h	Error estimate E_h
1	2.815440305995e+04	0.00000000	0.000000000000e+00
2	2.815129497153e+04	0.00000000	-2.072058951012e-01
3	2.815051550969e+04	3.98747994	-5.196412223668e-02
4	2.815032047025e+04	3.99643187	-1.300262932151e-02
5	2.815027171259e+04	4.00018017	-3.250510917618e-03
6	2.815025952307e+04	3.99996736	-8.126343595601e-04
7	2.815025647588e+04	4.00024493	-2.031461505491e-04
8	2.815025571409e+04	4.00002268	-5.078624963062e-05
9	2.815025552364e+04	4.00009486	-1.269626130428e-05
10	2.815025547603e+04	3.99996378	-3.174094066101e-06

Figure 35: Result of table of method rk4 using trapezoidal rule.

8.5 Matlab code for a3length

```
% Script which uses MyRichardson to get an approximation
    length using the
% trapezoid rule.
%
% PROGRAMMING by  Mathias Hallberg (c19mhg@cs.umu.se)
%                 Gustaf Soderlund (et14gsd@cs.umu.se)
%
%   2022-01-13  Finished the program

% Clean up
clear all

% Load parameters describing shot
a3f3

% Set initial time step
h0=1;

% Methods
m=["rk1","rk2","rk3","rk4"];

% Number of rows in table
kmax=10;

% Define the function needed for arc length
g=@(z)sqrt(z(3,:).^2+z(4,:).^2);
% Loop over methods
for i=1:4
    % Select method
    method=m(i);

    % Initialize time step
    dt=h0;

    % Initialize maxstep
    maxstep=200;

    % Loop over approximations
    for k=1:kmax
      % Compute range
      [r, flag, t, tra]=range_rkx(param,v0,theta,method,
          dt,maxstep);
      % Save information
      a(k)=a3int(g,t,tra);
      % Decrease time step
      dt=dt/2;
      % Increase maxstep
      maxstep=maxstep*2;
    end

    % Run Richardsons techniques
    data=MyRichardson(a,i);
```

```

    % New figure
    h(i)=figure();

    % Print to screen
    rdifprint(data,i);
end
```

9 Conclusion

Richardson's techniques can be applied when the user wants to make sure that the data is reliable by getting an error estimate in more advanced calculations.

Richardson's techniques provides an accurate approximation of a targeted value together with error estimations. We found the assignment to be challenging but at the same time interesting when we learned more about the techniques. We understand now that using these techniques it can take a lot of time in more advanced calculations, but that's a sacrifice to make for a more accurate approximation of the targeted value.