Georgia Journal of Science

Volume 79 *No. 3 Notes, Editorials, Letters, and Errata* Article 1

2021

Erratum: A NON-LINEAR APPROXIMATE SOLUTION TO THE DAMPED PENDULUM DERIVED USING THE METHOD OF SUCCESSIVE APPROXIMATIONS [Georgia Journal of Science, Vol. 76, No. 2, Article 9]

Javier E. Hasbun
University of West Georgia, jhasbun@westga.edu

Erratum: A NON-LINEAR APPROXIMATE SOLUTION TO THE DAMPED PENDULUM DERIVED USING THE METHOD OF SUCCESSIVE APPROXIMATIONS [Georgia Journal of Science, Vol. 76, No. 2, Article 9]

Acknowledgements

The author thanks K. Johannessen's private communication of the error referred to in the present work.

Erratum: A NON-LINEAR APPROXIMATE SOLUTION TO THE DAMPED PENDULUM DERIVED USING THE METHOD OF SUCCESSIVE APPROXIMATIONS [Georgia Journal of Science, Vol. 76, No. 2, Article 9]

Javier E. Hasbun*
University of West Georgia
Carrollton, Ga 30118

*Corresponding author
E-mail: jhasbun@westga.edu

ABSTRACT

In an earlier publication [Hill and Hasbun, 2018] considered an approximate solution to the damped pendulum, named the improved modified method of successive approximations (IMMSA), and compared it to an approximation from the work of [Johannessen, 2014]. Here, a correction is made to that comparison due to an error made in calculating Johannessen's approximation.

Keywords: Pendulum, successive approximation, damped pendulum, analytic solution, numerical solution, matlab, octave

INTRODUCTION

The damped pendulum system differential equation considered by Hill et al. [Hill and Hasbun, 2018] is

$$\frac{d^2\theta}{dt^2}+\frac{c}{m}\frac{d\theta}{dt}+\omega_0^2\sin\theta=0 \tag{1}$$

which can be written as

$$\frac{d^2\theta}{dt^2}+2\gamma\frac{d\theta}{dt}+\omega_0^2\sin\theta=0 \tag{2}$$

where $\omega_0=\sqrt{\frac{g}{L}}$, $\gamma\equiv\frac{c}{2m}$. Here g is the acceleration due to gravity, L is the pendulum's string length, m is the hanging mass at the end of the string, and c is the coefficient due to friction.

Figure 4, in the Hill et al. work, compares two approximations against the MATLAB [MathWorks] numerical solution of the above Equation (1). One of the approximations is what we called the improved modified method of successive approximations (IMMSA) whose results are given by Equations (34-47) [Hill and Hasbun, 2018]. The other approximation is that from Johannessen's work [Johannessen, 2014] and whose approximate solution to the above Equation (1) is given by Equations (57-58) [Hill and Hasbun, 2018] with the correction that Equation (58) should instead read

$$\xi(u) = (1 + \frac{1}{4}m(u) + \frac{9}{64}m(u)^2)u + \frac{1}{8\gamma_j}(m(u) - m_0) + \frac{9}{256\gamma_j}(m(u)^2 - m_0^2) \qquad (58')$$

where we rewrite $u = \omega_0 t$, $m(u) = m_0 \exp(-2\gamma_j u)$, $\gamma_j \equiv \beta / \sqrt{\omega_0^2 - \beta^2}$, $\beta = \gamma$, and the rest of the approximation is as presented by Hill et al. [Hill and Hasbun, 2018]. Here we note that we have introduced the γ_j and the β because, between them, they are the culprit that caused the miscalculatuion. In other words, our γ of the above Equation (2) corresponds to β in Johannessen's work [Johannessen, 2014].

With these corrections, we redo the comparison between the IMMSA and Johannessen's approximation. The results are shown in Figure 1.

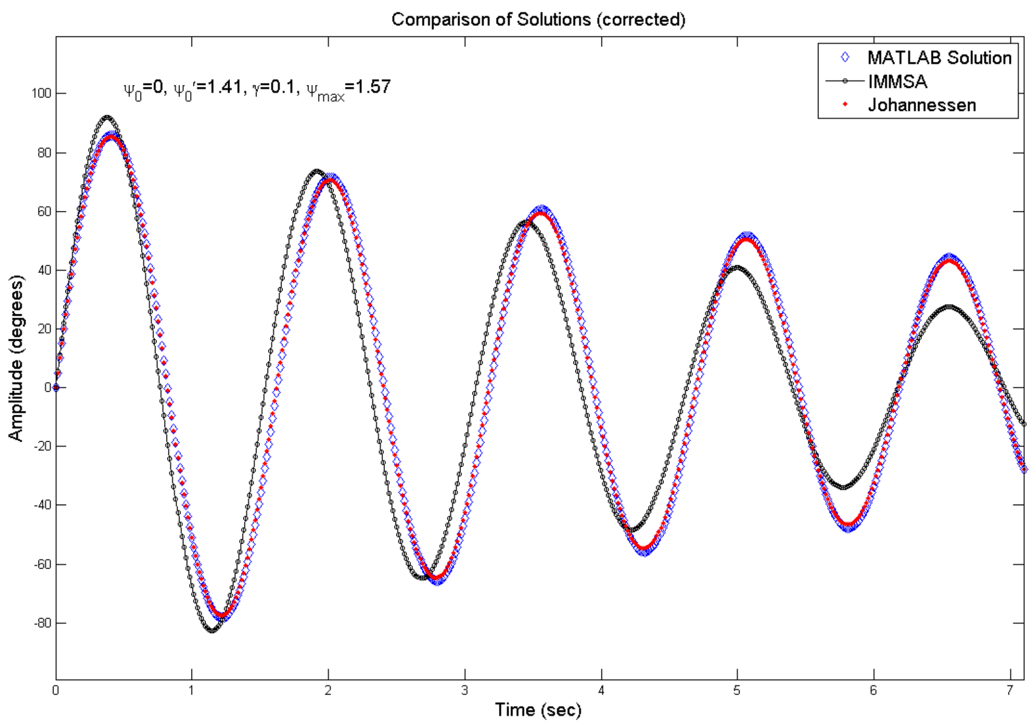

Figure 1: Corrected Graph produced by the script of the Appendix. It compares the IMMSA, Equations (25-27, 35, 36, 43, 47) and Equation's (57-58) of Johannessen's approximation in Hill et al [Hill and Hasbun, 2018] with the corrections made here (Equation 58') against MATLAB's numerical solution. The parameters used here are m=1, $\gamma = 0.1$, $c = 2m\gamma$, $\psi'(0) = 2sin(\psi_{max}/2)$, $\psi_{max} = \pi/2$, $\psi(0) = 0$ *[Johannessen, 2014].*

DISCUSSION

The purpose behind this paper is to correct an error made in a previous work [Hill and Hasbun, 2018] where Johannessen's approximation [Johannessen, 2014] was miscalculated due to the incorrect use of the parameter γ_j. Furthermore, in Hill et al., the accuracy of an approximation was determined by Equation 56 of that work, which for the IMMSA remains the same; that is, 0.2617, but the accuracy for the Johannessen's approximation is now 0.0213.

Finally, the appendix contains the corrected version of the MATLAB code used in obtaining the above Figure 1 with the corrections described here. This new code replaces the former one presented in Appendix C of our earlier work [Hill and Hasbun, 2018].

ACKNOWLEDGEMENTS

The author thanks K. Johannessen's private communication of the error referred to in the present work.

REFERENCES

Hill, Justin A. and Hasbun, Javier E. 8394092 (2018) *A Nonlinear Approximate Solution to the Damped Pendulum Derived Using the Method of Successive Approximations*. Georgia Journal of Science, Vol. 76, No. 2, Article 9.

Johannessen, K. *An analytical solution to the equation of motion of the damped nonlinear pendulum*. Eur. J. Phys. **35**, 035014 (2014).

MathWorks (www.mathworks.com) , owner of MATLAB. The in the paper code is also compatible with the open source Octave (https://www.gnu.org/software/octave/).

APPENDIX

This is the corrected code (corrects Appendix C of Hill et al. [Hill and Hasbun, 2018]) that is used to obtain the plot of the two approximate theoretical solutions, the IMMSA and Johannessen's (our Equation's (57-58) of Hill et al.) with the corrected Equation 58' of the present work. Both solutions are compared to the result of the MATLAB's ODE solver in Figure 1 above. The parameters used are as follows: m = 1.0, gamma=0.1, c=2*m*gamma, g = 9.8, L = 0.5, psi0=0.0, psi0'=2*sin(psi_max/2), psi_max =pi/2, and x = 0.6.

---------------- Script Listing ---------------

```
% IMMSA_and_Johannessen_corrected.m by  J. E. Hasbun (3/2021)
% This compares the IMMSA, the MATLAB solver, and Johannessen's
% solutions.
% This solves the full pendulum with damping numerically using a MATLAB
% solver as well as solving the approximate form through the
% by the improved modified method of successive approximation (IMMSA) which
% is compared to the work of Johannessen (Eur. J. Phys, V38, 035014
% (2014)).

function IMMSA_and_Johannessen
clear
global w0 m c

m=1.0;
t0=0.0;
g=9.8;
L=.5;
gam=0.1;                          %as used by Johannessen
c=2*m*gam;
B=c/(2*m);
cf=2*pi/360;                      %conversion factor from degrees to radians
w0=sqrt(g/L);
tau0=2*pi/w0;                     %period for the SHO
tmax=5*tau0;                      %maximum time

%Here are the conditions when psi_max and psi0 are provided
psi_max=pi/2;                     %maximum angle needed  - radians
psi0=0.0;                         %initial angle psi=thr - radians
psi0_p=2*sin(psi_max/2);          %initial psi prime=dtheta0/w0 - radians/sec
thr=psi0;                         %radians
dtheta0=w0*psi0_p;                %rad/sec
th=thr/cf;                        %theta_0 in degrees
NPTS=500;
dt=tmax/(NPTS-1);
t=[0:dt:tmax];

                     %The IMMSA solution
x=0.6;      %as used here

thr0=-1.4;  %For the amplitude of the IMMSA, for this comparison
y = fzero(@(y) y_iter(y,x,thr0,dtheta0),1.0);  %solve for y
A12=y*x*thr0;
A22=y*(1-x)*thr0;
om12=sqrt(w0^2*(1-A12^2/8)-B^2);
om22=sqrt(w0^2*(1-A22^2/8)-B^2);
del=atan(-(dtheta0+B*thr0)*(A12+A22)/(thr0*(om12*A12+om22*A22)));

%solve for t00 so that theta passes through zero at t=0 in this comparison
ff=@(tt) A12*cos(-om12*tt+del)+A22*cos(-om22*tt+del);
t00=fzero(@(tt) ff(tt),-1.5);
fprintf('thr0=%4.5f, y=%4.5f, t00=%4.5f\n',thr0,y,t00)
thIMMSA=exp(-B*abs(t-t00)).*(A12*cos(om12*(t-t00)+del)+A22*cos(om22*(t-
t00)+del));

                     %The Numerical Solution (MATLAB SOLVER)
```

```
ic1=[thr;dtheta0];
[tm,th2m]=ode45(@fderivs,[t0:dt:tmax],ic1);% matlab numerical solution
Error_thIMMSA=sqrt(sum((thIMMSA(:)-th2m(:,1)).^2)/NPTS);

                      %Johannessen's solution
beta=gam;                         %Johannessen's beta is our gamma
gam_j=beta/sqrt(w0^2-beta^2);  %Johannessens gamma (gamma_j here)
sc1=w0/sqrt(1+gam_j^2);
u=sc1*t;
mu0=(sin(psi_max/2))^2;
mu=mu0*exp(-2*gam_j*u);
xi=(1+mu/4+9*mu.^2/64).*u+(mu-mu0)/gam_j/8+9*(mu.^2-mu0^2)/gam_j/256;
[sn,cn,dn]=ellipj(xi,mu);
thJohann=2*atan(sqrt(mu).*sn./dn);
Error_thJohann=sqrt(sum((thJohann(:)-th2m(:,1)).^2)/NPTS);

fprintf('Error_thIMMSA=%4.5f,
Error_thJohann=%4.5f\n',Error_thIMMSA,Error_thJohann)

plot(tm,th2m(:,1)/cf,'bd');  %The MATLAB solver solution
hold on
plot(t,thIMMSA/cf,'ko-','MarkerSize',3);
plot(t,thJohann/cf,'r.')

legend('MATLAB Solution','IMMSA','Johannessen');
str=cat(2,'\psi_0=',num2str(psi0,3),', \psi_0\prime=',num2str(psi0_p,3),...
    ', \gamma=',num2str(gam,3),', \psi_{max}=',num2str(psi_max,3));
text(0.5,max(thIMMSA/cf)*(1+0.1),str);
axis([0 tmax min(thIMMSA/cf)*(1+0.2) max(thIMMSA/cf)*(1+0.3)])
xlabel('Time (sec)');
ylabel('Amplitude (degrees)');
title('Comparison of Solutions');

function fyzero=y_iter(y,x,thr,thrd)
global w0 m c
A1=y*x*thr;
A2=y*(1-x)*thr;
B=c/2/m;
om1=sqrt(w0^2*(1-A1^2/8)-B^2);
om2=sqrt(w0^2*(1-A2^2/8)-B^2);
fyzero=y-sqrt(1+((thrd+B*thr)/(om1*x+om2*(1-x))/thr)^2);

function derivs = fderivs(t,z)
global w0 m c
% pend2_der: returns the derivatives for the pendulum's full solution
% The function pen2_der describes the equations of motion for a
% pendulum. The parameter w0, is part of the input
% Entries in the vector of dependent variables are:
% x(1)-position, x(2)-angular velocity
derivs = [z(2); -w0^2*sin(z(1))-c*z(2)/m]; %the damping case is included now
```