

Análise Matmática II

Engenharia Informática Curso-Europeu

Atividade 02- Métodos Numéricos para a resolução de Sistemas de Equações Diferenciais

Ano letivo – 2021/2022

Grupo:

Bruno Guimar: a2021137345@isec.pt

Carolina Veloso: a2021140780@isec.pt

Luís Duarte: a2021137789@isec.pt

Docente:

Arménio Correia: armenioc@isec.pt

**Índice**

1. **Introdução**
2. **Desenvolvimento**
   1. SED
   2. Métodos para o SED
      1. Método de Euler
      2. Método de Euler Melhorado
      3. Método RK2
      4. Método RK4
      5. Solução Exata
3. **Problemas de Apllicação**
   1. Problema do pêndulo
   2. Problemas da mola-massa com amortecimento
   3. Problema da mola-massa sem amortecimento
   4. Problema do Modelo Vibratório Mecânico
   5. Problema do Circuito Elétrico em Série
4. **Conclusão**
5. **Referências**

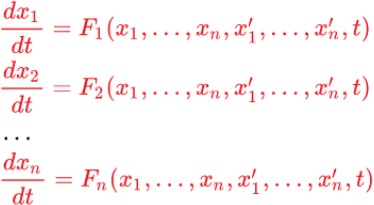
# Introdução

Para sermos capazes de resolver problemas complexos temos de ser capazes de os representar e estudá-los e para isso necessitámos de usar um sistema de equações diferenciais. Este trabalho visa ser capaz de entender como usamos Métodos numéricos de resolução de sistemas de equações diferenciais. Para isso iremos usar o método de Euler, o método de Euler Melhorado e o método Runge-Kutta de 2ª e 4ª ordem. Com a implementação deste trabalho no MATLAB, também seremos capazes de resolver vários problemas, como por exemplo, o problema do pêndulo.

# Desenvolvimento

# SED

Um sistema de equações diferenciais é um sistema que consiste em duas ou mais equações envolvendo derivadas de duas ou mais variáveis dependentes em relação a uma única variável independente.



# 

# Métodos para o SED

# Método de Euler

O método de Euler é um método numérico de primeira ordem para resolver equações diferenciais ordinárias com um determinado valor inicial. Nisso no caso, usaremos este método para resolver um sistema de equações.

function [t,u,v]= NEulerSED (f,g,a,b,n,v0,u0)

h = (b-a)/n;

t = a:h:b;

u = zeros (1,n+1);

v = zeros (1,n+1);

u(1) = u0;

v(1) = v0;

for i=1:n

u(i+1) = u( i ) + h\*f ( t ( i ), u ( i ), v ( i ) );

v(i+1) = v( i ) + h\*g ( t ( i ), u ( i ), v ( i ) );

end

end

* + 1. **Método de Euler Melhorado**

O método de Euler melhorado é semelhante ao método de Euler tradicional, a única diferença é que este método usa uma média das inclinações em cada ponto para cada iteração, dando assim maior precisão.

function [t,u,v]= Neuler\_Melhorada (f,g,a,b,n,v0,u0)

h = (b-a)/n;

t = a:h:b;

u = zeros (1,n+1);

v = zeros (1,n+1);

u(1) = u0;

v(1) = v0;

for i=1:n

uFun = f ( t ( i ), u ( i ), v ( i ) );

vFun = g( t ( i ), u ( i ), v ( i ) );

u(i+1) = u( i ) + h\*( uFun + f ( t (i+1), u (i) + h\*uFun, v (i) + h\*uFun ) ) /2;

v(i+1) = v( i ) + h\*( vFun + g ( t (i+1), u (i) + h\*vFun, v (i) + h\*vFun ) ) /2;

end

end

## 

## Método RK2

É um método numérico com alguma precisão, em grande parte devido à sua fórmula que considera para cada iteração dois valores chamados geralmente por "k", onde o primeiro é a inclinação no início do intervalo, a segunda é a inclinação no final do intervalo.

function [t,u,v]= NRK2SED (f,g,a,b,n,v0,u0)

h = (b-a)/n;

t = a:h:b;

u = zeros (1,n+1);

v = zeros (1,n+1);

u(1) = u0;

v(1) = v0;

for i=1:n

k1u = h \* f ( t ( i ), u ( i ), v ( i ) );

k1v = h \* g( t ( i ), u ( i ), v ( i ) );

k2u = h \* f ( t ( i+1 ), u ( i ) + k1u, v ( i ) + k1v );

k2v = h \* g( t ( i+1 ), u ( i ) + k1u, v ( i ) + k1v );

u(i+1) = u ( i ) + (1/2) \* (k1u + k2u);

v (i+1) = v ( i ) + (1/2) \* (k1v + k2v);

end

end

## 

## Método RK4

É o método numérico de maior precisão. A sua fórmula considera para cada iteração quatro valores normalmente chamados de "k", onde o primeiro é a inclinação no início do intervalo, a segunda é a inclinação no início do intervalo ponto médio do intervalo usando o primeiro declive, o terceiro é novamente o declive no ponto médio do intervalo, mas desta vez usando a segunda inclinação e finalmente a quarta é a inclinação no final da pausa. e equações.

function [t,u,v]= NRK2SED (f,g,a,b,n,v0,u0)

h = (b-a)/n;

t = a:h:b;

u = zeros (1,n+1);

v = zeros (1,n+1);

u(1) = u0;

v(1) = v0;

for i=1:n

k1u = h \* f ( t ( i ), u ( i ), v ( i ) );

k1v = h \* g( t ( i ), u ( i ), v ( i ) );

k2u = h \* f ( t ( i ) + ( h/2 ), u ( i ) + ( k1u/2 ), v ( i ) + ( k1v/2 ) );

k2v = h \* g( t ( i ) + ( h/2 ), u ( i ) + ( k1u/2 ), v ( i ) + ( k1v/2 ) );

k3u = h \* f ( t ( i ) + ( h/2 ), u ( i ) + ( k2u/2 ), v ( i ) + ( k2v/2 ) );

k3v = h \* g ( t ( i ) + ( h/2 ), u ( i ) + ( k2u/2 ), v ( i ) + ( k2v/2 ) );

k4u = h \* f ( t ( i ) + h, u ( i ) + k3u, v ( i ) + k3v );

k4v = h \* g ( t ( i ) + h, u ( i ) + k3u, v ( i ) + k3v );

u (i+1) = u ( i ) + (1/6) \* ( k1u + 2\*k2u + 2\*k3u + k4u );

v (i+1) = v ( i ) + (1/6) \* ( k1v + 2\*k2v + 2\*k3v + k4v );

end

end

## 

## Solução Exata

A solução exata diz exatamente o resultado que o exercício dado retorna, porque essa é a solução exata, muitas vezes usada como modo de comparação com outros métodos para tentar descobrir qual deles é o mais eficaz e preciso.

function [t, exata]= SolExata( ODE,a,b,n,v0,u0)

syms y(t);

tempExata = dsolve (ODE, [‘y(0)=‘, num2str(u0)], ...

[‘Dy(0)=’, num2str (v0) ], ‘t‘);

if (~isempty(tempExata) )

ext=@(t) eval( vectorize( char( tempExata) ) );

h=(b-a)/n;

t=a:h:b;

exata=ext(t);

else

exata= [ ];

end

# 

# Problemas de Aplicação

# Problema do Pêndulo

* Como este problema é uma equação diferencial homogênea não linear de ordem 2, a solução exata não será apresentada.
* O programa apresentará a função dada na lista, parâmetros, dados de entrada, métodos para resolver o problema, gráfico e uma tabela.
* Usando todos os métodos para fazer uma comparação entre eles, podemos ver a precisão de cada um para resolver este problema.

Graphical user interface, application

Description automatically generated

Chart, line chart

Description automatically generated

Table

Description automatically generated

## Problema Mola-Massa Com Amortecimento

* Este problema é um exemplo clássico para sistemas mecânicos e são utilizados para estudos de oscilações.
* Sabemos que a mola tem um comportamento linear, o amortecedor também e as superfícies em contacto não têm atrito.

Graphical user interface, application

Description automatically generated

Chart, line chart

Description automatically generated

Table

Description automatically generated

## Problema Mola-Massa Sem Amortecimento

* Neste caso, o processo será o mesmo, mas com uma condição importante, não existe amortecimento.

Graphical user interface, application

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated

## 

## Modelo Vibratório Mecânico

## Diagram Description automatically generated

## Graphical user interface, application Description automatically generated

## Chart, line chart Description automatically generated

Table

Description automatically generated

## Circuitos Elétricos em Série

* Podemos também determinar soluções de equações de diferencias de problemas como o problema do circuito elétrico, usando mais uma vez os métodos acima descritos.

Graphical user interface, application

Description automatically generated

Chart, line chart

Description automatically generated

Table

Description automatically generated

# 

# Conclusão

Com a elaboração deste trabalho, concluímos que os métodos numéricos funcionam também com sistemas de equações diferenciais, no entanto para conseguirmos resolver esses sistemas, precisamos de adaptar os métodos uma vez que tanto o input como o output vão ser diferentes. Percebemos também que se tivermos uma equação diferencial não linear, não existe maneira de termos uma solução exata. Como não temos uma solução exata, tivemos de adaptar o nosso código para que se for introduzida uma equação diferencial não linear, não é mostrado o output da Solução Exata uma vez que o retorno dessa função é um array vazio.

1. **Referências**

Data da última consulta: 4/06/2022

Moodle: <https://moodle.isec.pt/moodle/mod/forum/discuss.php?d=33674>

Youtube:<https://www.youtube.com/watch?v=BrYM8D8BUwM&ab_channel=FicouMaisF%C3%A1cil>