

SOMMAIRE

I- Introduction

II- Résolution numérique

II.1- Méthode d'Euler

a-Présentation de l'algorithme utilisé

b-Représentation des solutions $x(t)$, $y(t)$, $z(t)$ en fonction des différentes valeurs de r

c-Représentation des solutions dans l'espace (x, y, z)

II.2- Méthode de Runge-Kutta

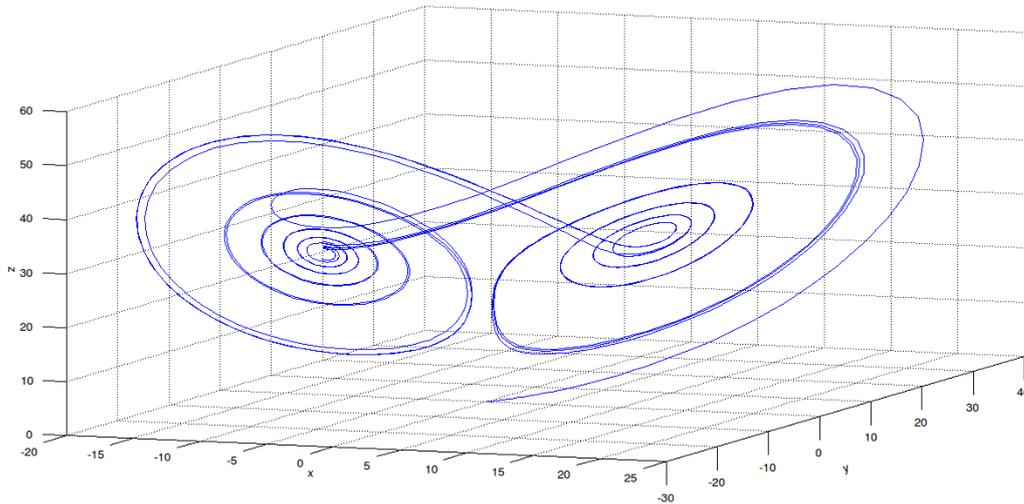
a-Présentation de l'algorithme utilisé

b- Représentation des solutions $x(t)$, $y(t)$, $z(t)$ en fonction des différentes valeurs de r

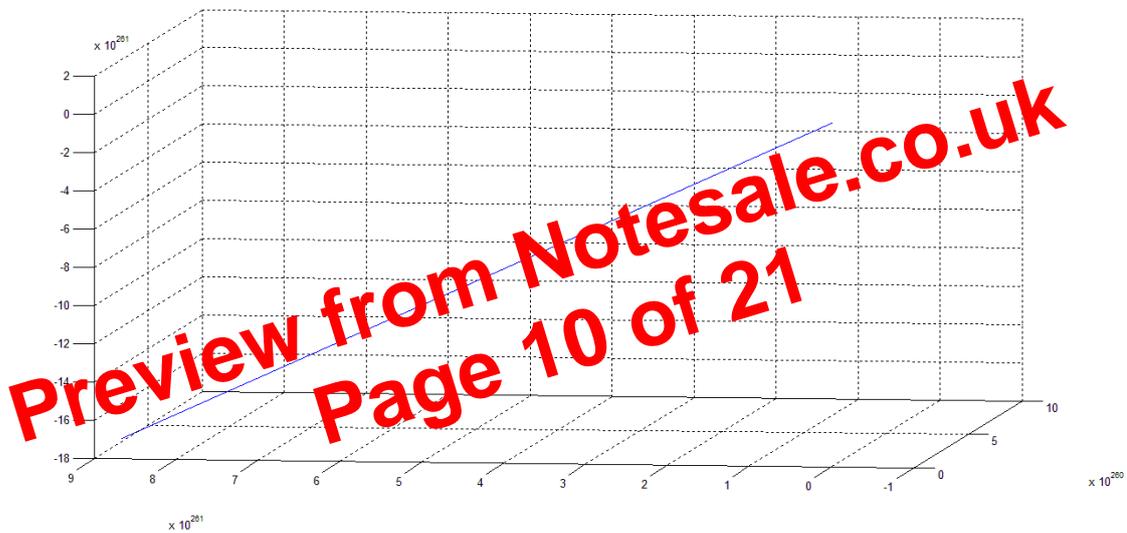
c-Représentation des solutions dans l'espace (x, y, z)

III- Etude des solutions en fonction des valeurs de r

IV- Conclusion



Solution $r=30$



Solution $r=180$

Le logiciel utilisé ici n'affiche rien quand r prend la valeur 180. En effet pour $r=180$, les résultats sont tellement proche de zéro que arrivé à la 62ème itération on a un résultat infini puis à partir de la 63ème on obtient des divisions par 0. Le résultat ci-dessus est obtenu grâce à Matlab qui manifestement est plus avancé.

II.1- Méthode de Runge Kutta

a-Présentation de l'algorithme utilisé

Pour des raisons de précisions, nous utilisons l'algorithme de Runge Kutta d'ordre 4 appliqué aux systèmes d'équations à conditions initiales. Voici l'algorithme. Dans l'optique d'avoir un meilleur confort et un algorithme optimal, nous avons écrit trois fonctions qui sont appelées dans le script ci-dessous.

```
h = 0.01;
```

```
t_h= [0.0:h:20];
```

```
t=0;
```

```
r=5;
```

```
for l = 1:1
```

```
for m = 1:2000
```

```
X1(1, 1) =0.001;
```

```
X1(l, m) =0;
```

```
end
```

```
end
```

```
for l = 1:1
```

```
for m = 1:2000
```

```
X2(1,1)=0.001;
```

```
X2(l, m) =0;
```

```
end
```

```
end
```

```
for l = 1:1
```

```
for m = 1:2000
```

```
X3(1, 1) =0.001;
```

```
X3(l, m) =0;
```

```
end
```

```
end
```

```
for l=1:3
```

```
for m=1:4
```

```
k(l,m)=0;
```

```
end
```

Preview from Notesale.co.uk
Page 11 of 21