

# 運動シミュレーション3

## 単振動

## Robotics入門

早大本庄 情報科 飯島 涼



# なぜ公式をグラフ化しない？

- グラフ化するための式を手計算しなくてはならない(本日の自由課題の例)
- そもそも式にするのが困難な運動がある
  - 運動に伴って、質量が変化する（雪だま）
  - 物体に加わる $F$ の値が時間によって変化するもの（次回、ロケット噴射）
- 公式を知らなくても、どのようにふるまうのか知らなくても初期値がわかれば試せる。（本日）

# 前回の内容

- 小数の誤差についての補足（エラーの原因その3）
- 公式で表しにくい運動をシミュレーションする
  - ロケットの逆噴射
  - 雨粒の落下

# 参考資料

- 大重美幸、詳細！Python 3 入門ノート、ソーテック社、2017
- 小高知宏、Pythonによる数値計算とシミュレーション、オーム社、2018
- 藤原邦男、基礎物理学 I 物理学序論としての力学、東京大学出版会、1984
- 山本義隆、駿台受験シリーズ 新・物理入門 増補改訂版、駿台文庫、1987
- 今井 功ほか、セミナーライブラリー物理学=2 演習力学[新訂版]、サイエンス社、1981

# float 型 (小数の型)

- int: 整数の型

- 文字 => 整数への変換 int( ) 関数

- float: 小数の型

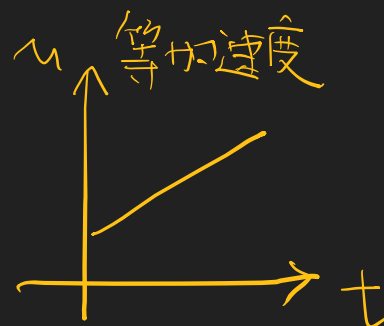
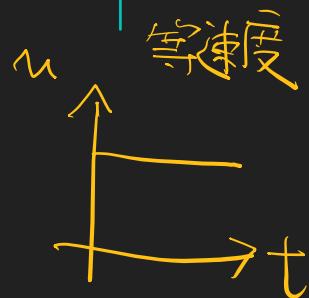
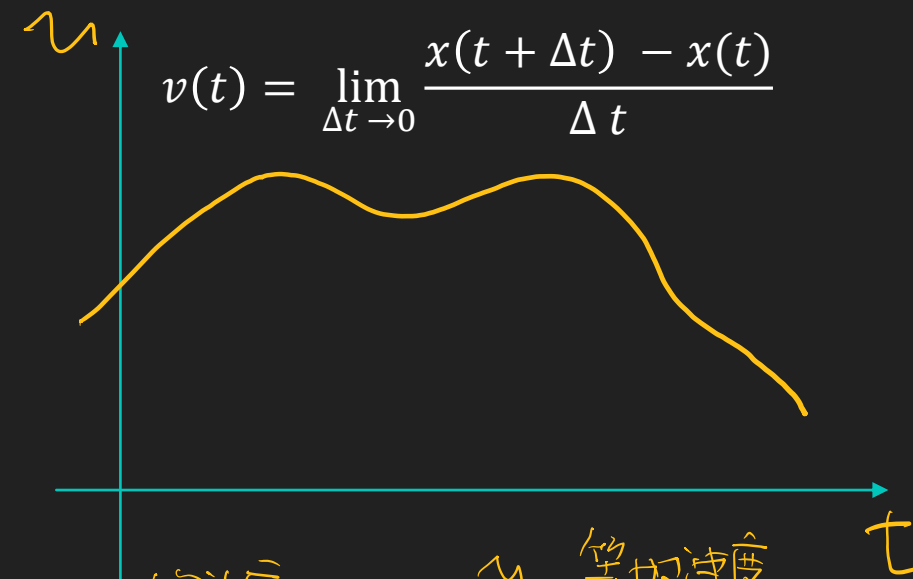
- 文字 => 小数への変換 float( ) 関数

- String: 文字の型

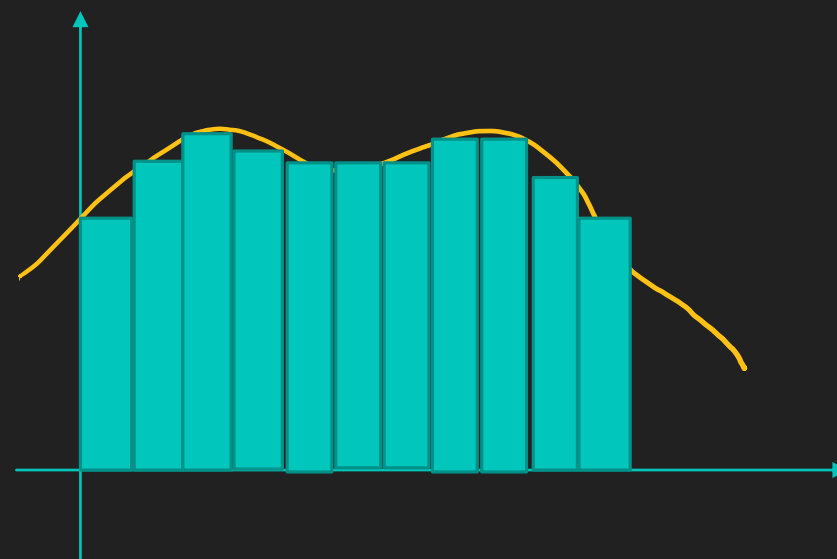
- 数字 => 文字への変換 str( ) 関数

# 物理の理論とプログラミングの違い

## ○ 物理（理論）



## ○ プログラミング



$\Delta t$ をある程度の小ささで妥協して、  
運動の振る舞いを観測してみる

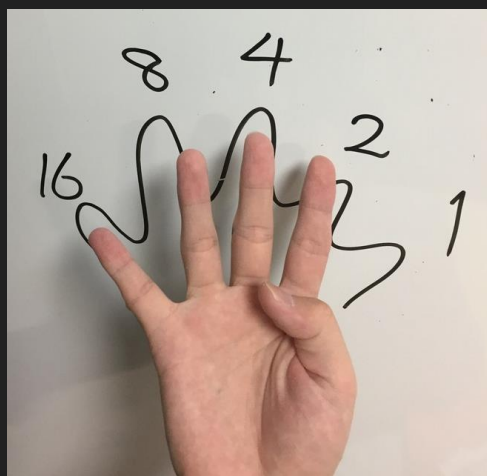
離散的なデータ

# 2進数

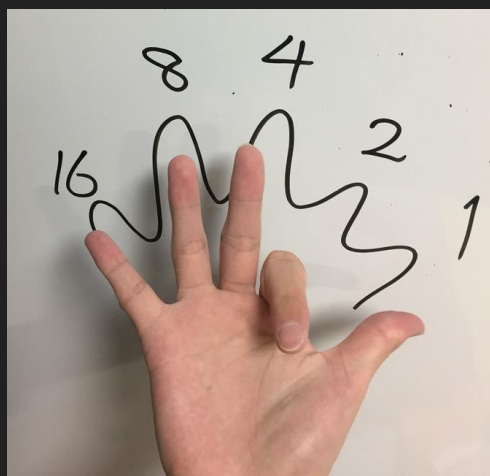
1	0	1	1
8 の 位	4 の 位	2 の 位	1 の 位

# 2進数の簡単数え方

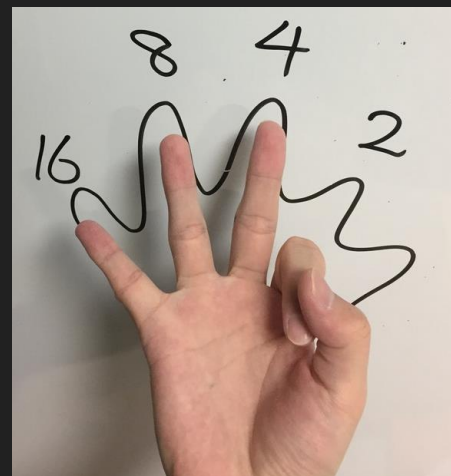
1



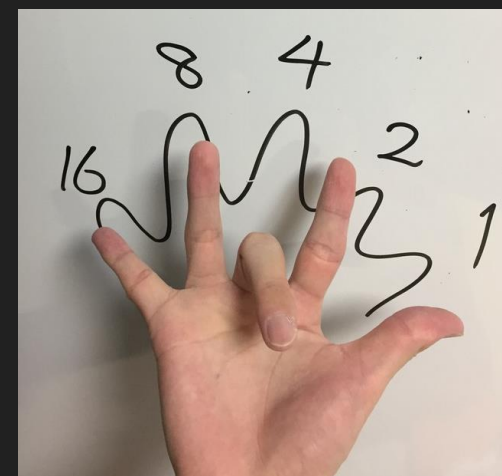
2



3



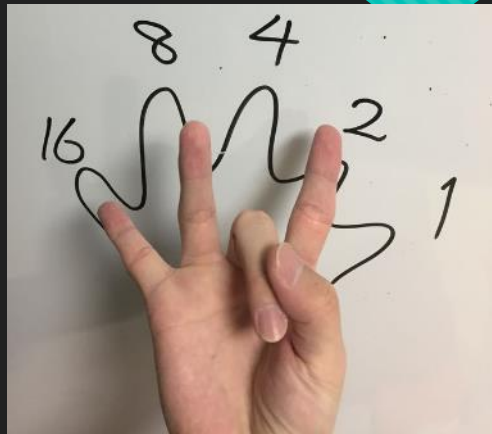
4



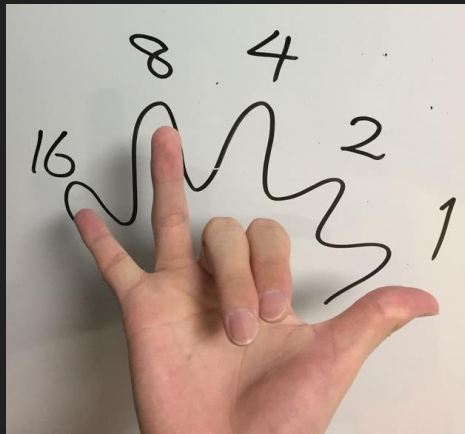


# 例続き

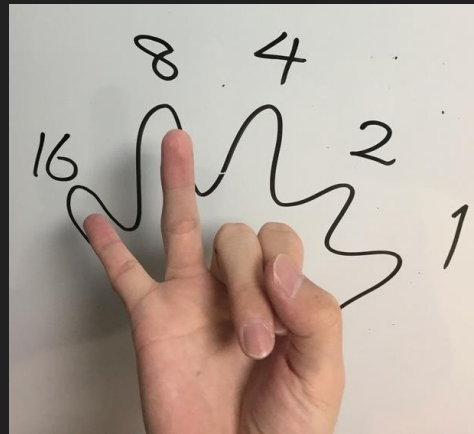
5



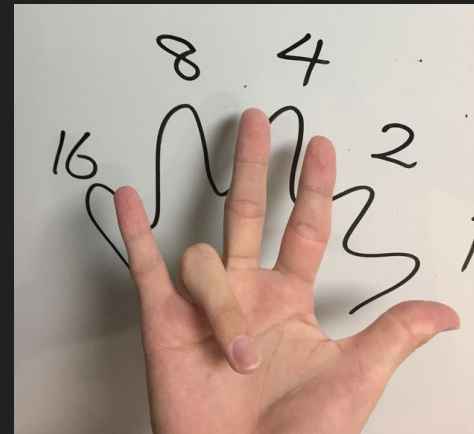
6



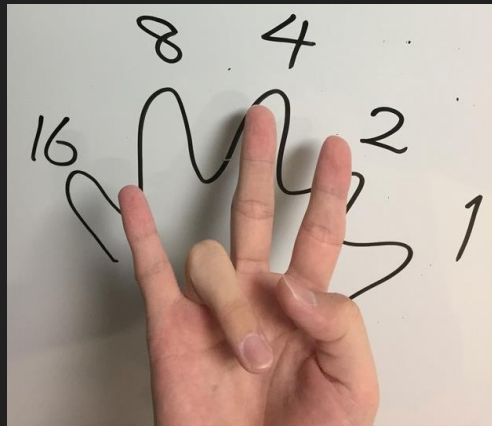
7



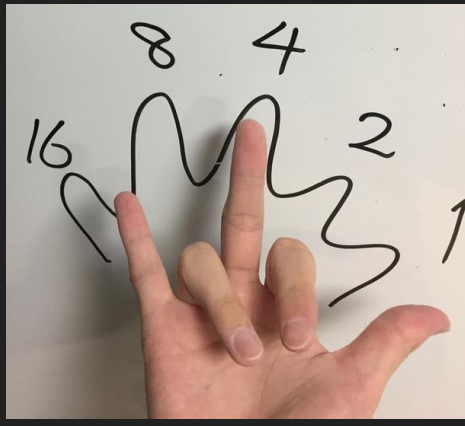
8



9



10



Q: この後の指の折り方はどうなるだろうか?

Q: いくつまで数えられそうか?

Q: 指の折り方に規則性はあるか?

# 2進数の小数

0	0	1	1	.	1	0	1	0
8	4	2	1		$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$
の	の	の	の		の	の	の	の
位	位	位	位		位	位	位	位



3 . 625

# 2進数の小数 コンピュータが普段していること

0	0	1	1	.	1	0	1	0
8	4	2	1		$1/2$	$1/4$	$1/8$	$1/16$
の	の	の	の		の	の	の	の
位	位	位	位		位	位	位	位



3.625

入力された10進数を2進数に変換してメモリに格納  
=>  $1/2^n$  の和でうまく表せない小数はどうなる？

# 小数と無限小数

## ○ Python での小数の表し方

$$a \times 2^n$$

a: 1. XXXXXXXXXXXXの形で表記した二進数

n: 指数部

aとnをペアで保存して小数を表す

ex)

$$1.00 \times 2^2 = 1 \times 2^2 = 4$$

⇒  $1/2^n$  の和でうまく表せない小数はどうなる?

⇒ 近似するしかない

# バグの原因 誤差

- 0.06 秒経過後に何か処理をしたい場合

```
1 t = 0.0
2
3 dt = 0.01
4
5 for i in range(50):
6     t += dt
7     print(t)
8
9     if (t == 0.06):
10         print("hit")
```

```
0.01
0.02
0.03
0.04
0.05
0.060000000000000005
0.07
0.08
0.09
0.09999999999999999
0.10999999999999999
0.11999999999999998
0.12999999999999998
0.13999999999999999
```

# 改善案（本日使います）

- 意図的に結果を四捨五入する（切り捨てもOK）



```
1 t = 0.0
2
3 dt = 0.01
4
5 for i in range(50):
6     t += dt
7     print(t)
8
9     if (round(t, 2) == 0.06):
10         print(round(t, 2))
11         print("hit")
```

```
0.01
0.02
0.03
0.04
0.05
0.06000000000000000005
0.06
hit
0.07
0.08
0.09
```

# 演習

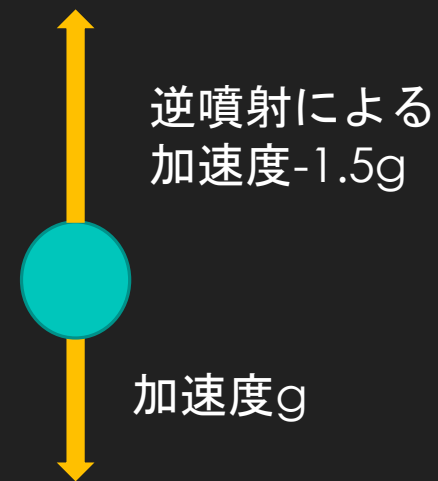
## ○ ロケットの逆噴射

- 初期速度 $v_0$ , 初期高度 $x_0$ , 逆噴射の開始時刻 $t_f$ をユーザが指定して、ロケットの着陸シミュレーションを行う

逆噴射前  $t < t_f$



逆噴射開始後  $t > t_f$



# 演習のプログラム例

<https://colab.research.google.com/drive/1aS643l6jy8FU0j4OqSYqtEbKIZHI46DT?usp=sharing>

```
1 #-*- coding: utf-8 -*-
2 # 9/17 用演習プログラム
3 import numpy as np
4 import matplotlib.pyplot as plt
5
6 # 定数
7 g = 9.8
8
9 # tf 秒後に逆噴射の加速度を返す関数
10
11 def retrofire(t, tf):
12     if t >= tf:
13         return -1.5 * g #重力の1.5倍の|加速度で逆噴射を行う（ロケットの重さは考えない）
14     else:
15         return 0.0;
16
17 t = 0.0 # 時刻t
18 dt = 0.01 # 時刻の刻み幅
19
20 # 係数の入力
21 v = float(input("初速度v0を入力してください:"))
22 x0 = float(input("初期高度x0を入力してください:"))
23 tf = float(input("開始時刻tfを入力してください:"))
24 x = x0 # 初期高度の設定
```

時間経過後に、逆噴射分の加速度を返す関数



## 続き（前回の復習+a）

```
27 tlist = [t]
28 xlist = [x]
29 # 自由落下の計算
30 while (x > 0) and (x <= x0):
31     t += dt                                # 時刻の更新
32     # ここに記入                            # 速度の計算
33     # ここに記入                            # 位置の更新
34
35     tlist.append(t)
36     xlist.append(x)
37 # グラフの表示
38 plt.plot(tlist, xlist) # グラフをプロット
39 plt.show()
```

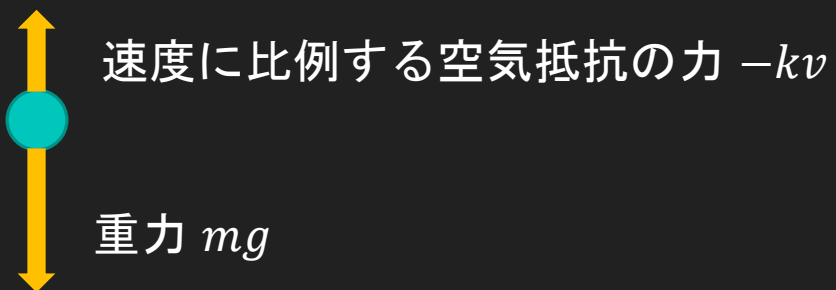
# できたかなと思ったら

- $v_0 = 0$ ,  $x_0 = 100$ ,  $t_f = 2.62$  として、滑らかな高度のグラフが描かれるかどうか試してください
  - 自由落下と同様の軌道を描いた場合 => 着陸失敗
  - 初期高度  $x_0$  を超えてしまう場合 => 着陸できずに再び空へ飛び立ちます
- 上記の条件でうまくいったら、 $t_f$  や  $v_0$  などの値をいろいろ試して、どのような場合に着陸失敗するのか、空へ飛び立つのかを調べてみてください。

# 雨粒の運動

- 雨はどのようにして降ってくるのか？
  - 空で雨粒ができてから、重力で加速し続ける？
  - 何らかの力が働いて、遅くなる？
  - 何らかの力が働いて、一定になる？

# 雨粒にかかる力



Q: 加速度はどうなる？

# 運動方程式

$$ma = F$$

$$ma = mg - kv$$

$$a = g - \frac{k}{m}v$$

# 演習 シミュレーション

雨粒の重さ  $m = 0.0000005$  [kg]

空気抵抗の定数  $k = 1.089 \times 10^{-5}$  [ $kg \cdot m^{-1} \cdot s^{-1}$ ]

重力加速度  $g = 9.8$  [ $m \cdot s^{-2}$ ]

初期高度  $x_0 = 2000$  [m]

として、時間における雨水の速度の変化をグラフ化してください。

前回のプログラムを利用してください。

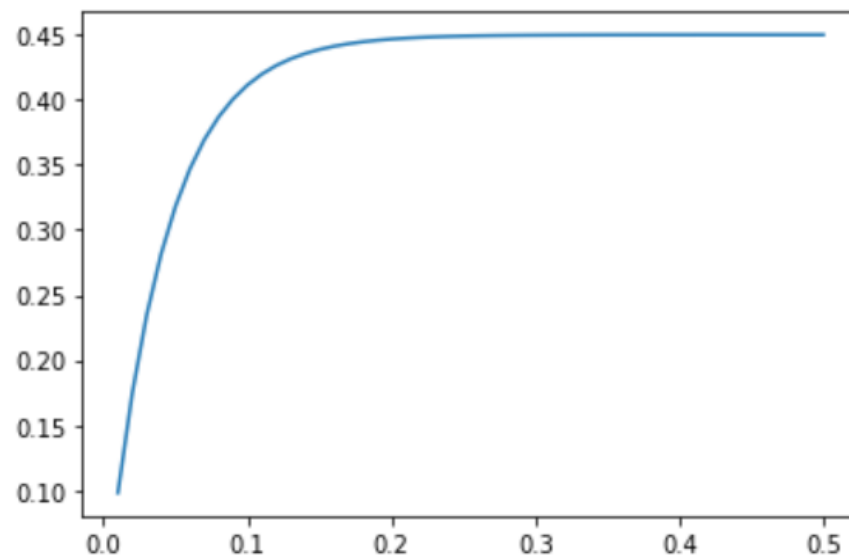
加速度の計算には、1ループ前の速度を利用可能であると仮定します。

ある程度たったら答えのグラフを見せます。

# 前回の演習こたえ + 訂正 + 終端速度

```
12 print(m*g / k) # 終端速度
13
14 tlist = []
15 vlist = []
16
17 while x > 0:
18     t += dt
19     a = g - k/m * v
20     v += a * dt
21     x -= v * dt
22
23     tlist.append(t)
24     vlist.append(v)
25
26 plt.plot(tlist[:50], vlist[:50])
27 plt.show()
28
```

終端速度: 0.4499540863177227



# 補足: 加速度の定義と、微分積分の関係

$$a(t) = \lim_{\Delta t \rightarrow 0} \frac{v(t+\Delta t) - v(t)}{\Delta t} = v'(t)$$

- 加速度も、速度と同様の議論で上の式を導ける(p. 3-9). (興味があれば考えてみてください)

The diagram illustrates the relationship between position  $x(t)$ , velocity  $v(t)$ , and acceleration  $a(t)$ . On the left, a blue bracket labeled "†で微分" (differentiate with †) groups the equations  $x(t) = \frac{1}{2}gt^2 + v_0t + x_0$ ,  $v(t) = x'(t) = gt + v_0$ , and  $a(t) = v'(t) = g$ . On the right, a yellow bracket labeled "†で積分" (integrate with †) points from the acceleration equation back to the velocity equation.

$$\begin{array}{lcl} x(t) & = & \frac{1}{2}gt^2 + v_0t + x_0 \\ \text{†で微分} \left\{ \begin{array}{l} v(t) = x'(t) = gt + v_0 \\ a(t) = v'(t) = g \end{array} \right. & & \left. \begin{array}{l} \text{†で積分} \end{array} \right\} \end{array}$$

Q(最終課題用): 加速度から積分して導いた後、積分定数をどのように計算すれば上の公式が導けるだろうか?(応用)



# 単振動

- 位置の変化が、三角関数( $\sin$ ,  $\cos$ )で表せる運動のこと
  - バネ
  - 振り子

# なぜ単振動をシミュレーションするのか？

X: 将来、バネや振り子を揺らして稼ぐような職業・仕事が存在する

## 応用例

- 「ロボットが目的の位置にたどりつくために、どの程度の力を加えればよいか」を考える
  - バネ・マス・ダンパモデル
    - 車両、乗り物全般で用いられているモデル（特に位置の計算が必要なもの）
    - 工学で用いられる機器の内部で利用されるモデル

# バネの運動 復習

## ○ バネにかかる力

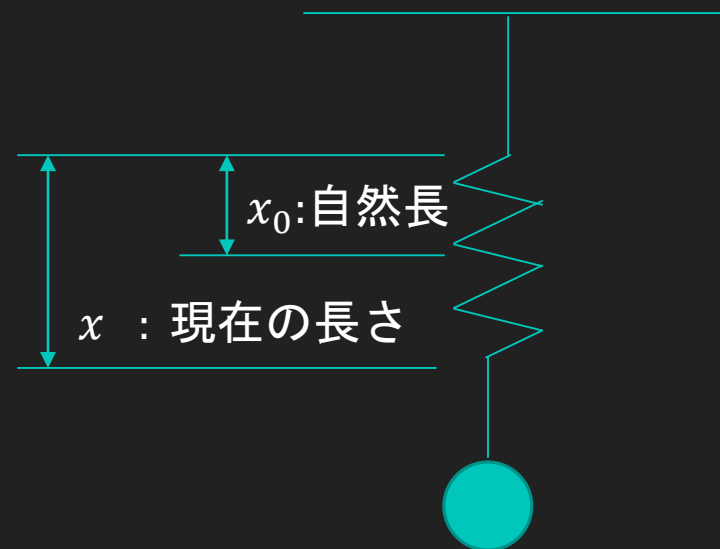
$$F = -k(x - x_0)$$

$= -k \times (\text{自然長からのバネの伸び})$

$k$ : バネ係数

自然長の位置を  $x_0 = 0$  と定義すると、

$$F = -kx$$



# バネの加速度

$ma = F$  に  $F = -kx$  を代入して、

$$ma = -kx$$

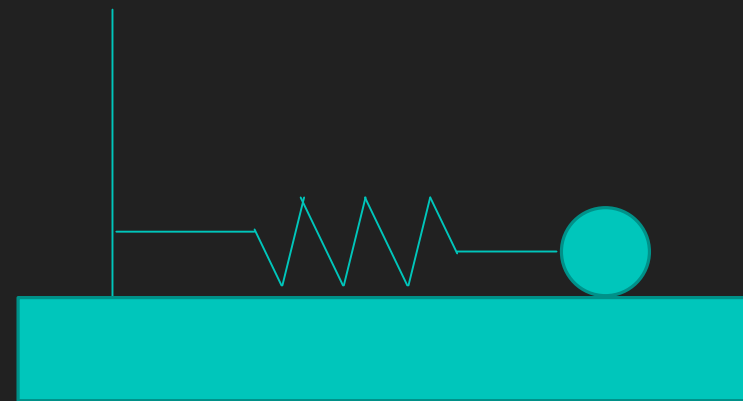
$$a = -\frac{k}{m}x$$

加速度のみをヒントに、バネの運動をシミュレーションしてみる【演習1】

# 【演習1】 単振動の運動シミュレーション

- バネの自然長の位置を $x_0 = 0$  mと定義する。バネ係数  $k = 5$  [N/m]のバネに、 $m = 3$  kg の重りを取り付けた後、バネを $x = 1$ mの長さになるように伸ばし、静かに離した。話を単純にするため、まずは地面と重りの間には摩擦が生じていないものと仮定する。
- 前ページで求めた加速度を利用して、バネの運動が三角関数で表せる並みの形になることを確認してください。

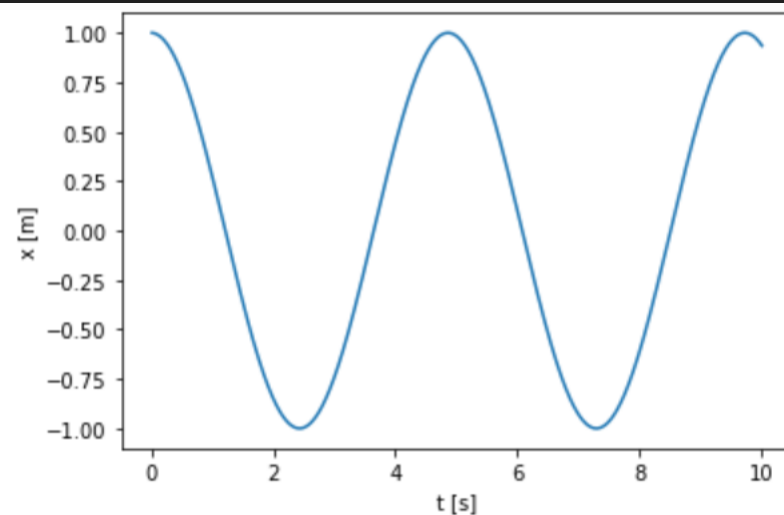
※ 前回まで用いていた運動シミュレーションのプログラムはバネなどの要因がかかわってきても同様に利用できる  
単振動の式を忘れていても/知らなくても再現ができる



# シミュレーション例

- 初期値の例: できた場合はバネ係数などを変えていろいろ試してみてください。

```
1 import matplotlib.pyplot as plt
2 m = 3 # 重りの重さ[kg]
3 x = 1 # 重りの初期位置[m]
4 k = 5 # バネ係数 [N/m]
5 x0 = 0 # バネの自然長の位置を原点とする。 [m]
6 t = 0.0 # 時間 t[s]
7 dt = 0.01 # 区切る時間の幅  $\Delta t$ 
8 v = 0 # 初速度 [m/s] 重りは静止した状態から始めてよい
```




# 単振動の運動についてわかったこと (仮定: 単振動の振る舞いについて何も知らない)

- 単振動の運動は以下の式で表せそうということが分かった (前スライド)

$$x = A \cos(\omega t + \theta)$$

$\omega$ : 角加速度 (一秒間にどのくらい回転する?),  $\theta$ : 位相 (どの角度からスタートする?)

角加速度の導出

微分		$x = A \cos(\omega t + \theta)$
		$v = -\omega A \sin(\omega t + \theta)$
微分		$a = -\omega^2 A \cos(\omega t + \theta) = -\omega^2 x$

演習1で求めた加速度

$$a = -\frac{k}{m}x$$

と、前ページの結果

$$a = -\omega^2 x$$

を比較して、

$$\omega^2 = \frac{k}{m}$$

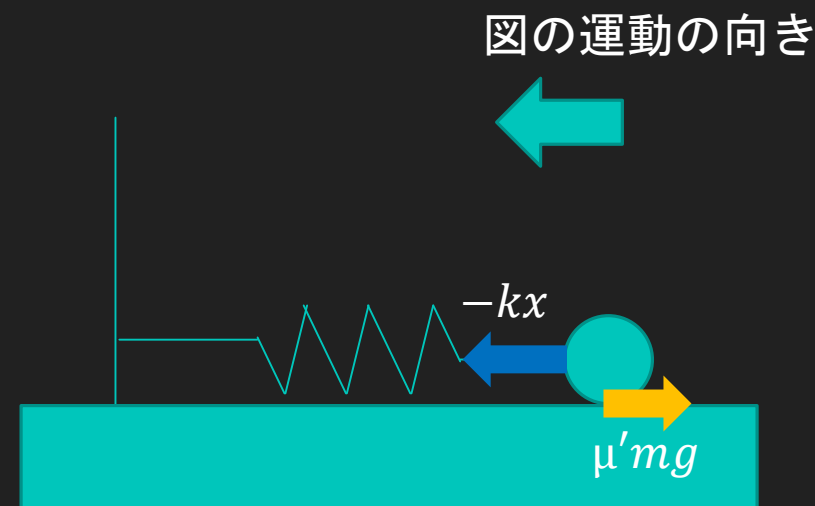
$$\omega = \sqrt{\frac{k}{m}}$$



## 【演習2】 地面との間に摩擦が生じる場合

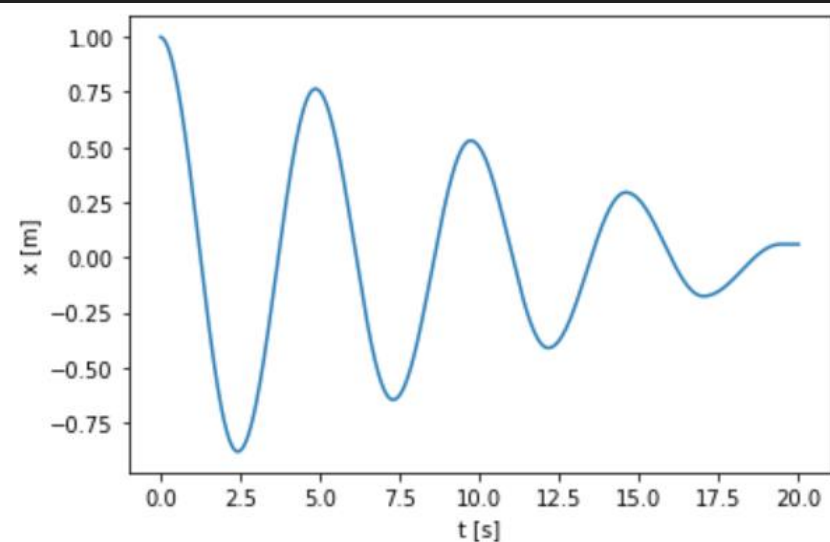
- バネの自然長の位置を $x_0 = 0$  mと定義する。バネ係数  $k = 5$  [N/m] のバネに、 $m = 3$  kg の重りを取り付けた後、バネを $x = 1$  mの長さになるように伸ばし、静かに離すと、静止することなく動き始めた。動摩擦係数を $\mu' = 0.01$  として、**運動と逆向きに動摩擦力が生じる場合の運動**をシミュレーションしてください。（動き始める前の静止摩擦力などは考慮しなくてよい。）
- 前回の運動からの**位置の変化**が、  
増えているか  
減っているか  
を判断すればよい。

演習1を少し書き換えれば動きます。



# 演習2 答え

```
1 import matplotlib.pyplot as plt
2 import numpy as np # ヒント: np.sign()を使用して、運動の向きを取り出す
3 m = 3 # 重りの重さ[kg]
4 x = 1 # 重りの初期位置[m]
5 k = 5 # バネ係数 [N/m]
6 x0 = 0 # バネの自然長の位置を原点とする。 [m]
7 t = 0.0 # 時間 t[s]
8 dt = 0.01 # 区切る時間の幅Δt
9 v = 0 # 初速度 [m/s] 重りは静止した状態から始めてよい
10 g = 9.8 #[m/s^2]
11 ud = 0.01 #  $\mu'$  の値
```



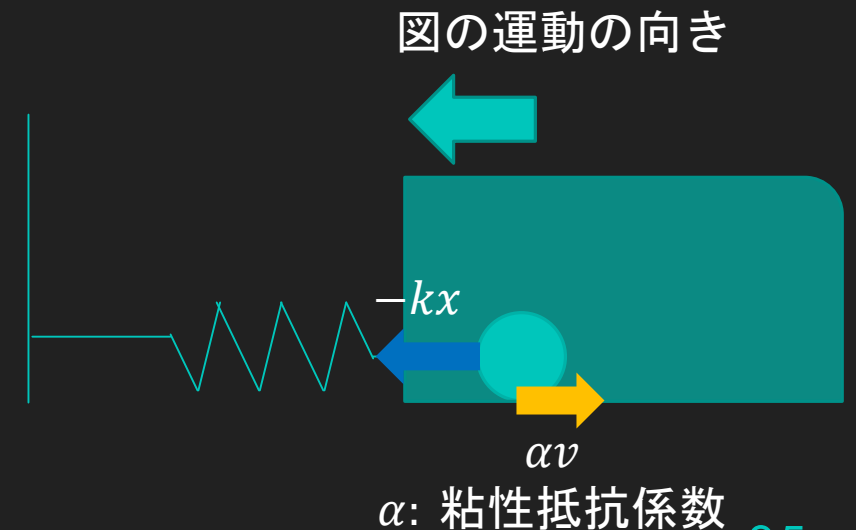
# 【演習3】 液体中のおもりの運動

おもりが水中を運動する場合（地面との摩擦力は0とする）

運動の方向と逆向きに、速度に比例する力がかかる（粘性抵抗、前回の空気抵抗と同様）

演習1, 2と同様の条件で、粘性抵抗係数 $\alpha = 0.01$  としたとき、  
（図参照）

運動はどのようなになるかグラフを作成して教えてください。



# 水の中のおもり運動 ヒント

```
1 import matplotlib.pyplot as plt
2 import numpy as np # ヒント: np.sign()を使用して、運動の向きを取り出す
3 m = 3 # 重りの重さ[kg]
4 x = 1 # 重りの初期位置[m]
5 k = 5 # バネ係数 [N/m]
6 x0 = 0 # バネの自然長の位置を原点とする。 [m]
7 t = 0.0 # 時間 t[s]
8 dt = 0.01 # 区切る時間の幅 $\Delta t$ 
9 v = 0 # 初速度 [m/s] 重りは静止した状態から始めてよい
10
11 alpha = 0.01 # 液体の粘性抵抗
12
13 TIME_END = 2000
```

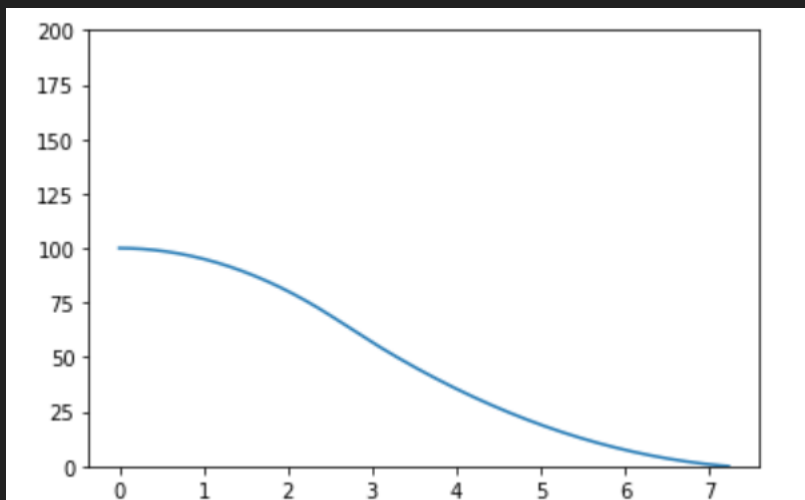
# 考察事項【演習2-3】

- 演習2のように、一定の力で抵抗力が働く場合と、演習3のように、速度に比例する形で抵抗力が働く場合、収束の仕方は同じか？異なるか？
  - 異なる場合、それぞれどのように収束するか？
- 演習3 の収束の仕方の概形を示すために効果的な式を導出 or 調べてください。
- バネの運動を工学的に応用する場合、どのような使い道が考えられるか、考えて or 調べてみてください。調べて出てきたものについては、適切に引用・参考文献を記入すること。あなたが考えた独創的なアイデアでも構わない。

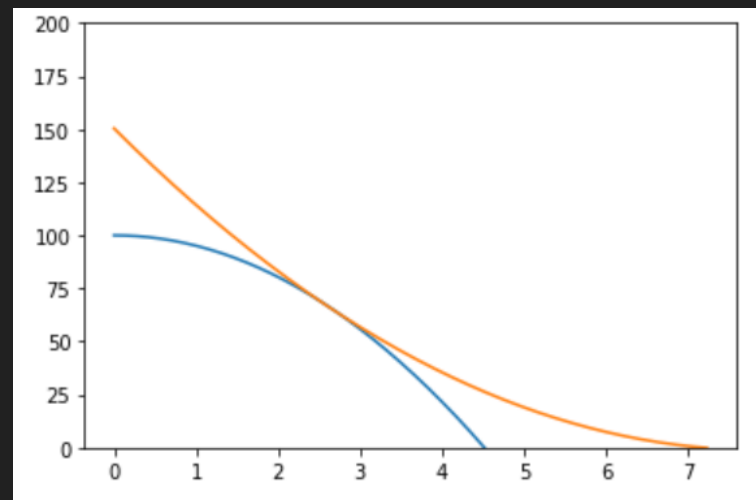
# 【演習4】 応用課題 「逆噴射の公式導出」

- ロケットの逆噴射で作成したコードに、高度 $x$ を表す式を導出し、シミュレーションで作られた軌道を再現する $t$ - $x$ グラフ を記述してください。（下の右側グラフが応用、やや難しいです、微積を利用）

シミュレーションで作ったグラフ



公式に当てはめて作ったグラフ



オレンジの線と青の線をつなぎ合わせると、シミュレーションの形と一致する。  
青の線: 自由落下の公式をグラフ化 オレンジの線: 逆噴射をしている場合の落下の公式 (?) をグラフ化

# 考察事項【演習4】簡単すぎた人むけ

- 物理で習った自由落下の式（前前回の授業で導出）は、どのような条件の下で成り立っているものだったのか？ 暗記しているときには気づかなかった、暗黙の了解となっている条件を詳細に述べて、習った自由落下の式が成り立たない場合の具体例を挙げてください。
- 演習1-3までで、あなたは本来の公式を知らない状態で単振動の運動を再現することに成功した。公式がなくても再現ができるならば、運動の形について公式を導出することにはどのような意味があるのだろうか？ 幾何学的な歴史（ユークリッド「原論」=> Fermat, Descartes, 解析幾何学）を調べてその理由を答えるか、現在の応用（ゲーム、CG、ロボットなど）に有用な理由を答えてください。  
hint: 吉田洋一ほか、数学序説 Math&Science (ちくま学芸文庫), 2013改訂 など
- 課題の感想を教えてください。

# レポート

- 演習1-4までのうち、できた部分のソースコードとグラフ、その説明を添付して、提出してください。テストが近いので締め切りは 10/16 (金) 23:59 (JST)としようかと思います。
- 2学期か3学期の成績に入れます（おそらく中間テスト程度の配分）。貢献・自由課題の双方に取り組んでいない方は、必ず提出をお願いします。