運動シミュレーション1
等速度運動・等加速度運動

早大本庄　情報科　飯島　涼

なぜプログラミングで運動?

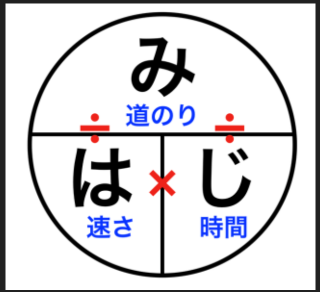
- ゲームプログラミング
- 工学全般
- ドローン・車・航空機の自動運転の制御
  - https://www.youtube.com/watch?v=q4Ugu4iJukM
- 車・航空機のシミュレーション（次回）

著作権画像

https://game.watch.impress.co.jp/docs/news/1274927.html

運動の復習

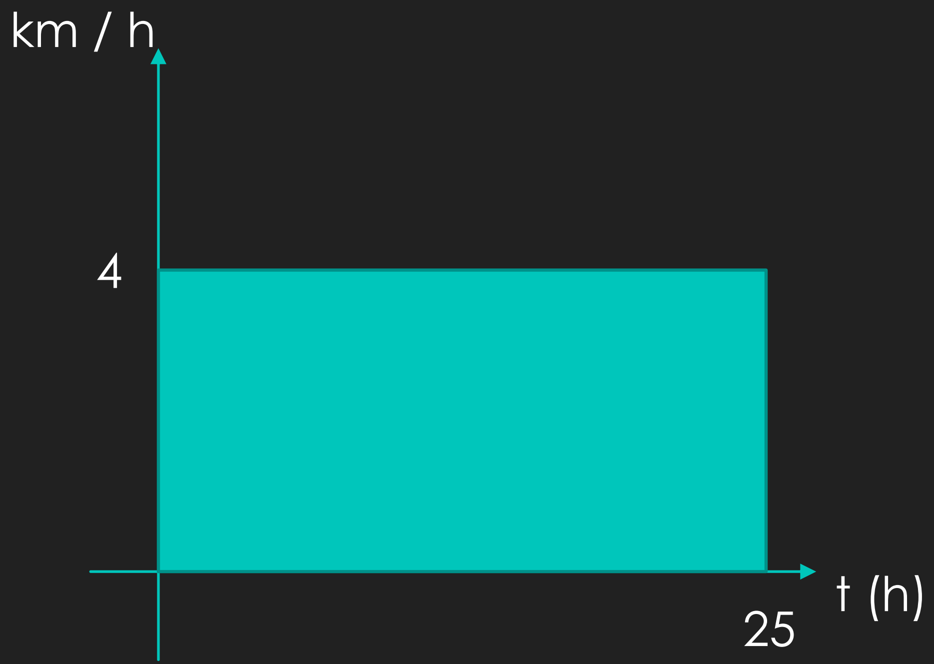
- 速さ = 道のり　÷　時間



- 100 km の道のりを，25 時間かけて走りました．

答え: 100 ÷ 25 = 4 km / h

$$\frac{100}{25}$$



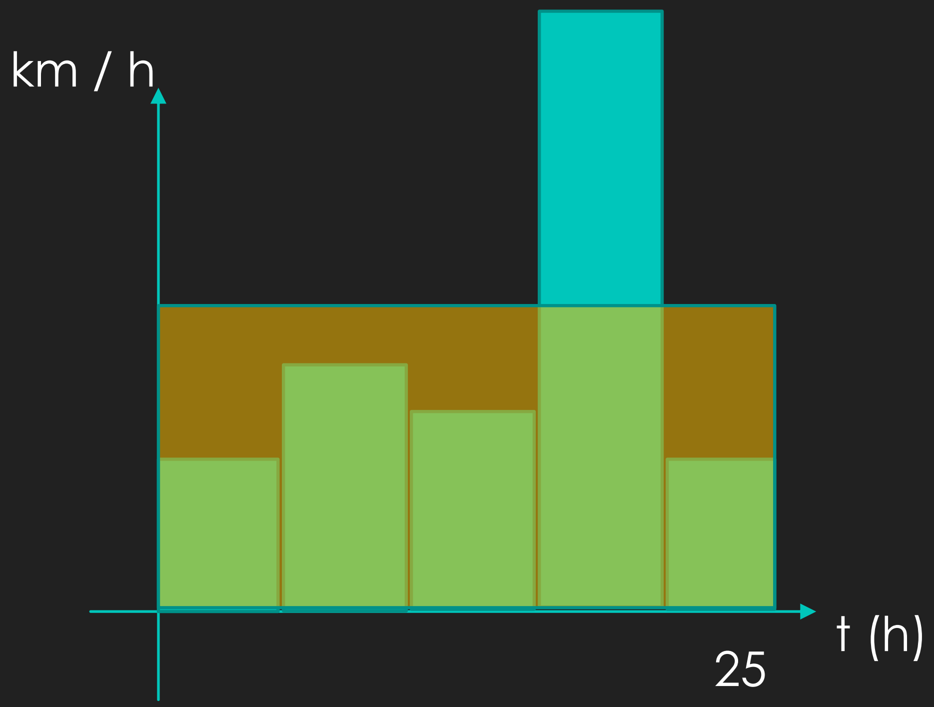
Q：小中学校で習った速さと，高校で習う速度って同じ?

どうすれば もっと詳細がわかる?

速さ: 平均速度のこと. 道のり ÷ 時間 で表される.

時間の幅を狭くする

- 最初の5時間で15km　$\frac{15}{5} = 3$
- 次の5時間で　20km
- 次の5時間で　10km
- 次の5時間で　50km
- 次の5時間で　 5km　$\frac{5}{5} = 1$



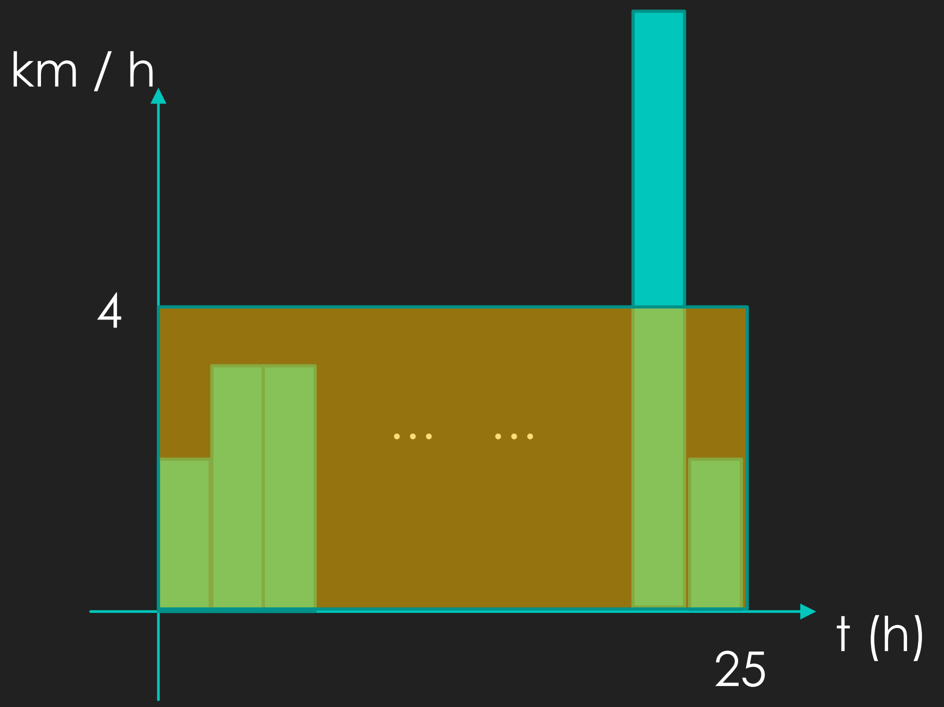
時間の幅を狭くする

- 最初の1時間で2km $\frac{2}{1}$
- 次の1時間で　3km
- 次の1時間で　3km $\frac{3}{1}$

…

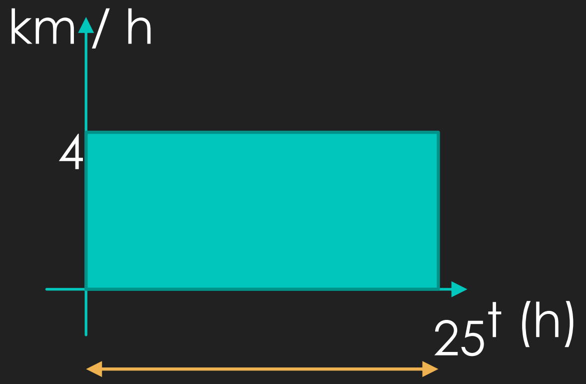
……

- 次の1時間で　20km $\frac{20}{1}$
- 次の1時間で　2 km

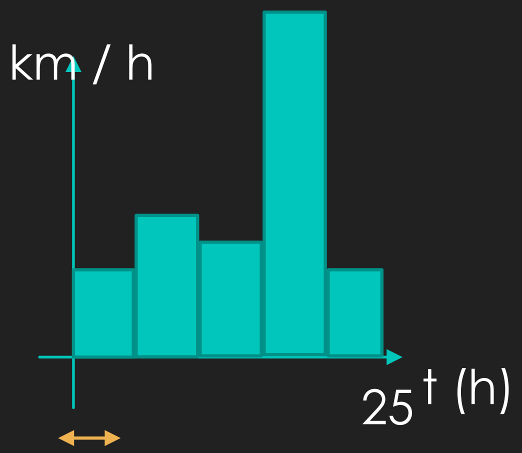


Δt: 区切る時間の幅

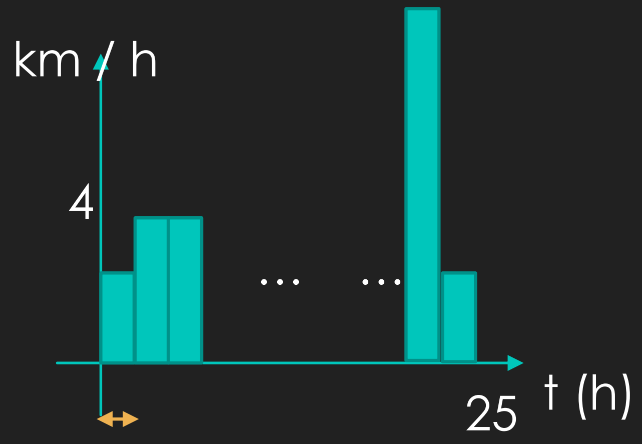
$\frac{100}{25}$ → $\frac{15}{5}$ → $\frac{3}{1}$



∇t = 25



∇t = 5



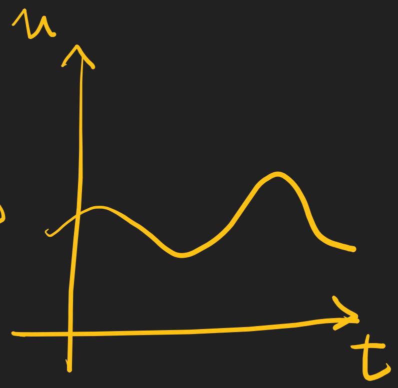
∇t = 1

速度とは

- 区切る時間幅Δt　が小さいほど，今この瞬間の速度が正確にわかる

Q: 区切る時間幅Δtをかぎりなく小さくしていったらどうなる？

$$\frac{100}{25} \Rightarrow \frac{15}{5} \Rightarrow \frac{3}{1}$$



時刻tでの位置を，$x(t)$とすると，

$$\lim_{\Delta t \to 0} \frac{x(t+\Delta t) - x(t)}{\Delta t} = v(t)$$

瞬間速度 v (m/s)

区切る時刻Δt を，限りなく小さくしたときの速さのこと

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

公式の復習

初期位置をx_0, 初速度をv_0, 重力加速度をgとしたとき，$x(t)$は,

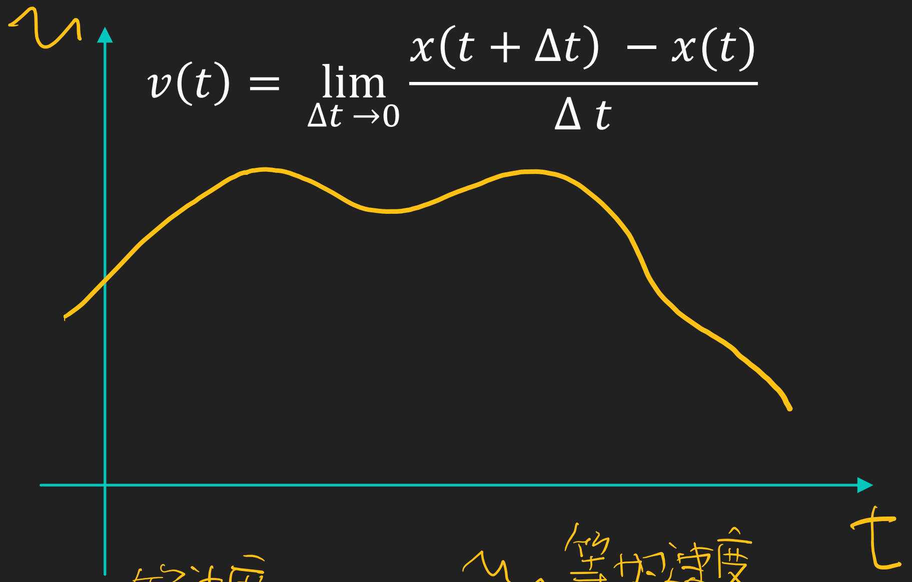
$$x(t) = \frac{1}{2}gt^2 + v_0 t + x_0$$

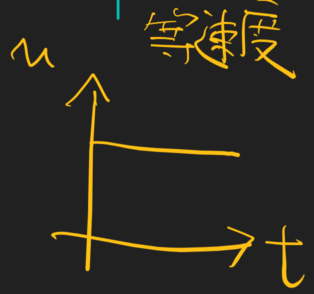
$$v(t) = x'(t) = gt + v_0$$

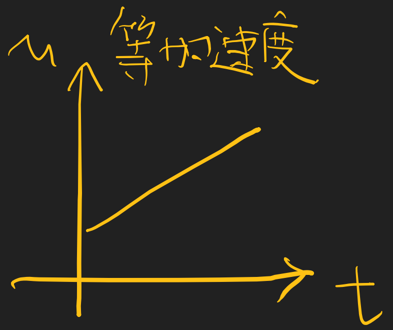
物理の理論とプログラミングの違い

- 物理（理論）

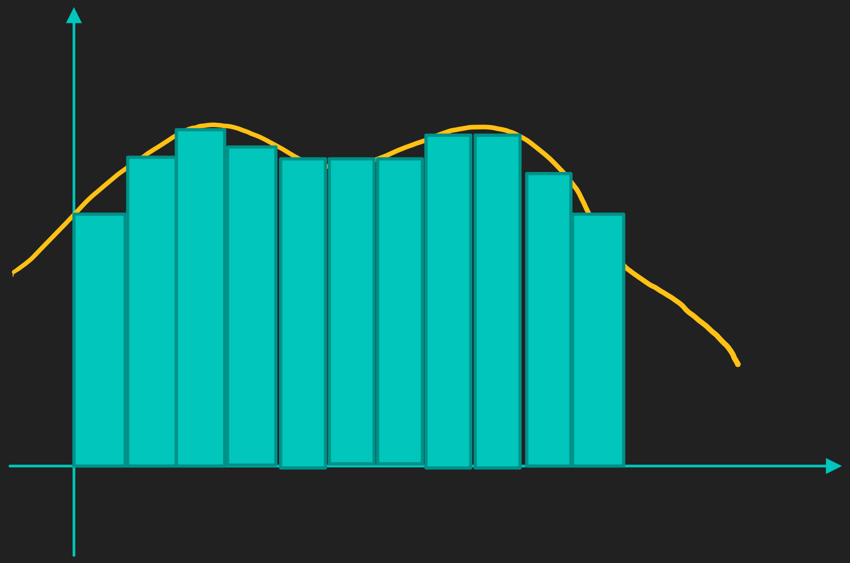
$$v(t) = \lim_{\Delta t \to 0} \frac{x(t+\Delta t) \ - x(t)}{\Delta t}$$







- プログラミング

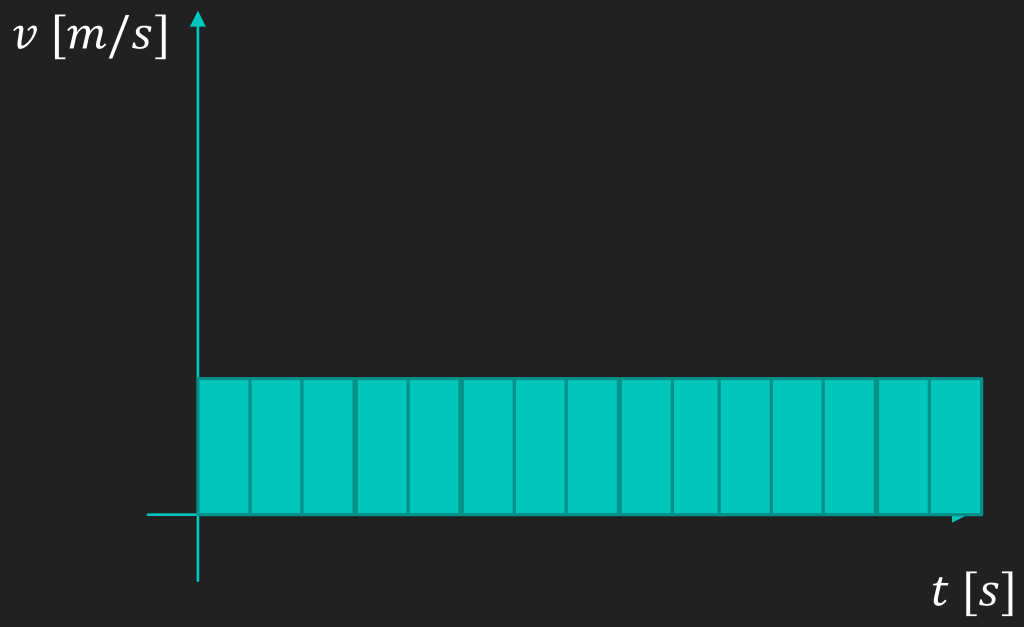


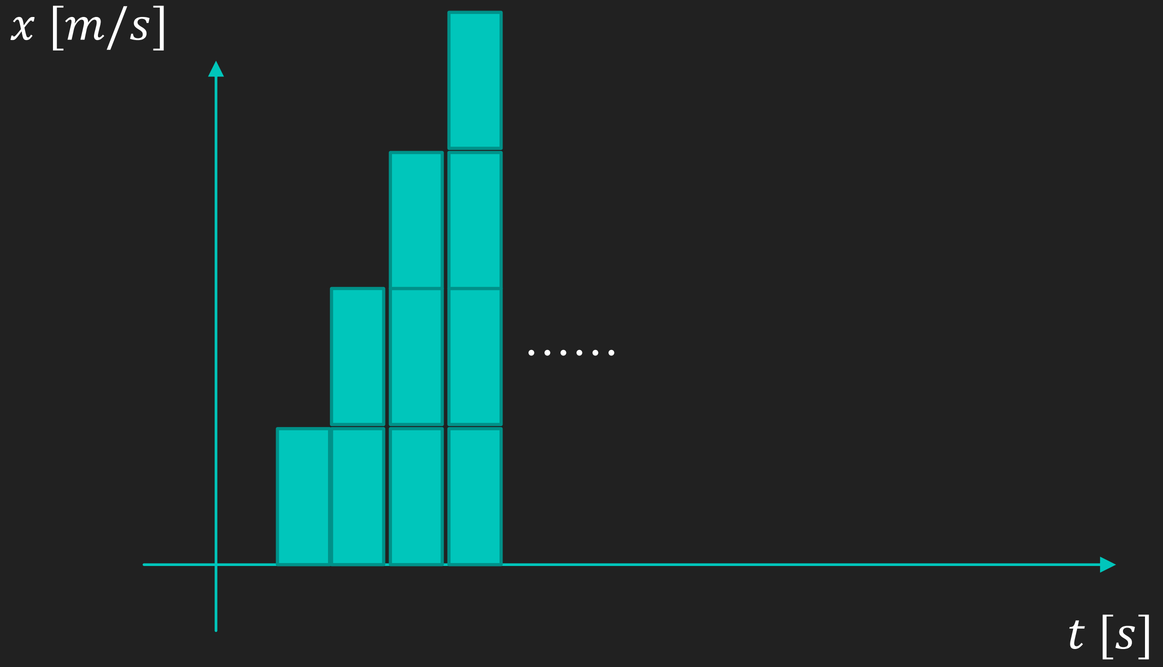
Δtをある程度の小ささで妥協して，
運動の振る舞いを観測してみる

離散的なデータ

運動シミュレーションの基礎

- 等速度運動





変数 v (固定値を代入)

変数 x (時間経過ごとに位置の変化を足していく)

プログラムと運動との対応

```
v0 = 10 #速度 v0 [m/s]
x0 = 0.0 # 位置 x0 [m]

t = 0.0 # 時間 t[s]
dt = 0.01 # 区切る時間の幅Δt

TIME_END = 20

while int(t) < TIME_END:
①  t += dt
②
```

0.01秒ごとの経過

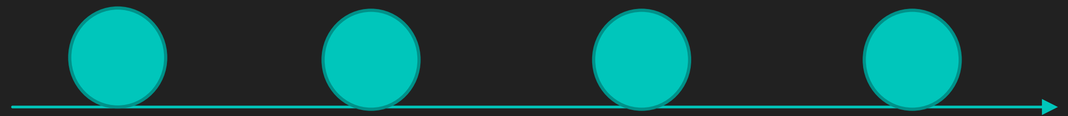
①

1回分の繰り返しを，Δt 秒の時間経過とみなす

②

Δt 秒時間が経過するごとに，その位置をリストに記録していく

=> ストロボ写真



演習 [等速度運動]

- 等速度運動となるように，オレンジの部分を埋めて下さい
- Δt ごとに，進む距離をxにどんどん足していく

ヒント: p3 にあった小学生用の図

プログラムのテンプレート配布先
https://colab.research.google.com/drive/1e6L0g74OSp9dmD5A8jOjKaG6HzESxfxu?usp=sharing

```
import matplotlib.pyplot as plt
v = 10 #初速度 v0 [m/s]
x = 0.0 # 初期位置 x0 [m]

t = 0.0 # 時間 t[s]
dt = 0.01 # 区切る時間の幅Δt

TIME_END = 20

tlist = []
xlist = []

while int(t) < TIME_END:
①   t += dt
②   # [x に, Δt の間に進んだ距離を足す]

    tlist.append(t)
    xlist.append(x)

plt.plot(tlist, xlist)
plt.show()
```

固定

時間ごとに位置を記録
=> ストロボ写真

加速度 $a[m/s^2]$

- プログラミング用の定義: 単位時間あたりに，速度に加える値のこと

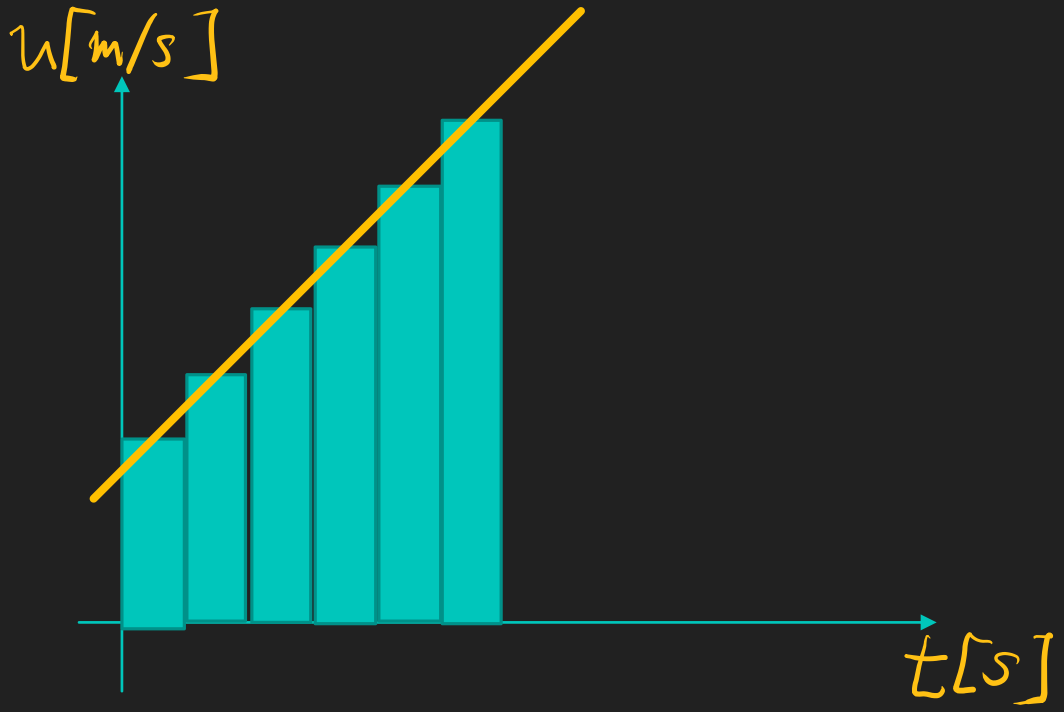
$\boldsymbol{a[m/s^2]}$: 1[s]あたり，a[m/s] を速度に加える

例)

$\mathbf{5}[\boldsymbol{m/s^2}]$:

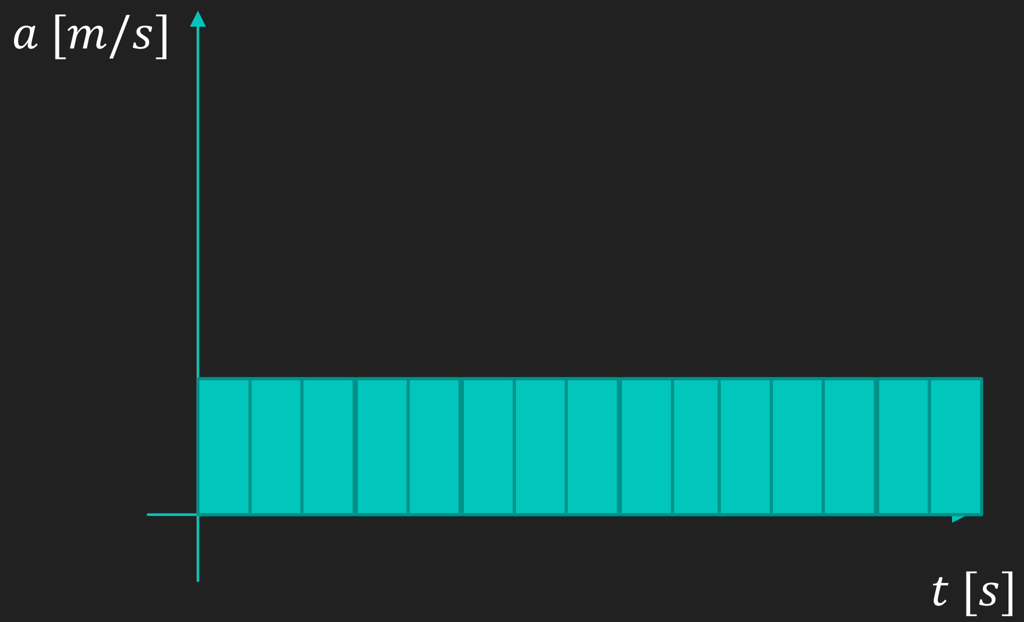
1[s]あたり，5[m/s] を速度に加える

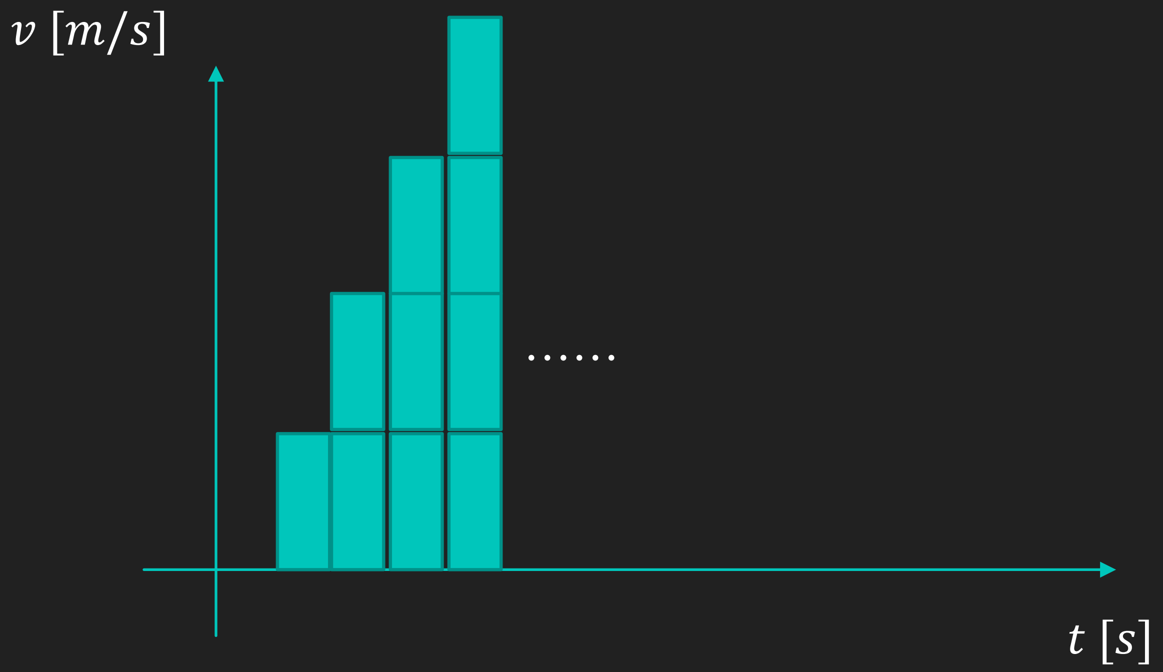
0.01[s] あたり，$0.01 \times 5\,[m/s]$



運動シミュレーションの基礎

- 等加速度運動





変数 a (固定値を代入)

変数 v (時間経過ごとに速度の変化を足していく)

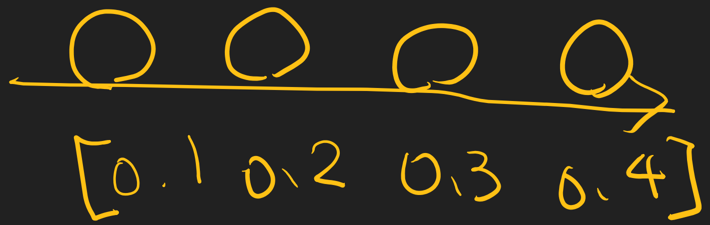
演習
[等加速度運動]

- 加速度が一定の運動
  - 例: 自由落下 g

```
import matplotlib.pyplot as plt
g = 9.8 # 重力加速度 g[m/s^2]
v = 10 #初速度 v0 [m/s]
x = 100 # 初期高度 x0 [m]

t = 0.0 # 時間 t[s]
dt = 0.01 # 区切る時間の幅Δt

TIME_END = 20

tlist = []
xlist = []

while x > 0:
  t += dt
  # [vに, Δt の間に増えた速度を足す]
  # [x から, Δt の間に落ちた距離を引く]

  tlist.append(t)
  xlist.append(x)

plt.plot(tlist, xlist)
plt.show()
```

シミュレーションまとめ

- オイラー法
  - Δtごとに時間を進めて，速度や位置を記録していくシミュレーション方法



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いけない
- 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.

自由課題

1. t=0で初期位置 x0, 初期速度 v0,質量mの物体に対して, 一定の力Fがかかり続ける運動について

   初期位置 x0, 初期速度 v0,質量m, 力Fをユーザから受け取り, 時間ごとの位置を示すグラフを表示するプログラムを作成してください. (t[s]-x[m]グラフ) TIME_END = 20秒でOKです.

2. t=0で初期位置 x0, 質量mの物体が初期速度 v0で等速度運動を行っているとき,

   一定の力Fがかかり始める時間 t_1 をユーザから受け取り, その運動の振る舞いを1. と同様にグラフ化するプログラムを作成してください. TIME_END = 20秒でOKです.

自由課題

3. 地面からの摩擦力や，落下時の空気抵抗を考慮するためには，演習で作成したプログラムをどのように書き換えればいいか，考えてください（次回説明）．可能であれば，自分で動摩擦係数などの条件を設定してプログラムを作成してみてください（提出はしなくてもいいです）

補足: 加速度の定義と，微分積分の関係

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

- 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）

tで微分

$$\begin{aligned} x(t) & & &= \frac{1}{2}gt^2 + v_0 t + x_0 \\ v(t) &= & x'(t) &= gt + v_0 \\ a(t) &= & v'(t) &= g \end{aligned}$$

tで積分

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)

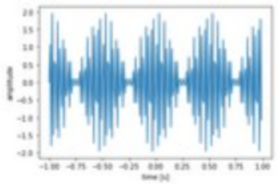
テスト解説

設問3（配点：20点）

物理の授業で，異なる周波数を出す2つの音叉（おんさ）を同時に叩いてうなりが発生する現象を確認したところ，「これは集団幻覚だ！」と主張するクラスメイトが現れたので，プログラムでうなりができている様子を客観的に示すことにした．
周波数 F1 Hz, F2 Hz で，各位相が0，振幅が1であるような、2つの正弦波を合成させたときの波形をグラフで出力し，うなりができることを確認してください．F1，F2 はユーザから入力として受け取れるようにすること．**グラフの横軸は時間とし，周波数の定義に合うように書くこと（軸の名前は省略可、単位が秒[s]でない場合はその旨を記載してください）．**

入力例
F1：440
F2：442

出力例
うなりの概形が確認できるグラフ



※入力例出力例を厳格に合わせなければいけないということはなく、表示などは自分なりにアレンジして良い。表示されるグラフがあっていれば良い。

解答例

```
import numpy as np
import matplotlib.pyplot as plt

F1=int (input("周波数 F1: "))
F2=int (input("周波数 F2: "))
A=1

t=np.arange(-1, 1, 1/100)
y=A* np.sin(2*np.pi*F1*t)+A* np.sin(2*np.pi*F2*t)

plt.xlabel("t")
plt.ylabel("y")
plt.plot(t, y)
```

設問4

流行りが廃れたにも関わらず行列が解消されないタピオカ屋に経営コンサルタントとして派遣されたあなたは，以下の要件でシミュレーションを行うことにした．
以下のシミュレーションを行う際、待ち行列を表現する特殊なライブラリを使用する必要はない。

問1 (配点:5点)

以下の要件で作成したシミュレーションを提出してください．

要件1: 1 秒あたりの客の到着 確率は， p= 0.05 であることがわかっている．
要件2: 客は，ランダムに 1 人ずつ到着する 一緒の来店は 1 人と カウント、整数時間ぴったりにしか来ないと考えてよい
要件3: for 文が 1 回まわるプログラムの動作を、 1 秒の経過とみなして、 TIME_END=360000 秒シミュレーションする。

図の要件1: 客の到着間隔をそれぞれリストに格納し，リストに格納された値を正規化されたヒストグラムとして表示してください．
図の要件2: また，そのヒストグラムの概形を表す数式を求め，ヒストグラムに重ねて表示してください．
授業内演習で作成したプログラムを流用して構わない．

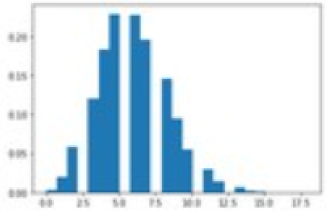
問2（配点：15点）

以下の要件を満たすシミュレーションを作成してください.
要件1: 要件2–要件 6までの内容を100000回繰り返す。(例: TIMES = 100000)
　要件2: num_person=0とする。
　要件3: 1 秒あたりの客の到着 確率は, p= 0.05 であることがわかっている.
　要件4: 客は, ランダムに 1 人ずつ到着する。一緒の来店は 1 人とカウント、整数時間ぴったりにしか来ないと考えてよい.
　要件5: 0秒からTIME_END=120秒までを問1と同様に シミュレーションする。
　要件6: 0秒からTIME_END秒まで何人の人が来たかを変数num_personで記録し、リスト(num_person_list)などに追加する。

図の要件1: 要件1–要件6までの処理でできたリストnum_person_listを正規化されたヒストグラムにしてグラフを出力して欲しい。

変数名は説明を簡単にするための例であり、自由に変えて構わない。
図の要件1を満たすグラフを作成してください。ユーザから受け取る入力などは一切必要ない。
ヒストグラムの棒の数は、ヒストグラムの概形が見てわかるように調節すること。

出力例:



```
TIMES = 0
num_person_list=[]
while TIMES <= 100000:
  time_dif = 0
  num_person=0
  time_end = 120
  time_dif_list = []

  for k in range(time_end):
    x = random.random()

    if (x < 0.05):
      time_dif_list.append(time_dif)
      time_dif = 0
      num_person +=1

    else:
      time_dif += 1

  num_person_list.append(num_person)
  TIMES +=1


plt.hist(num_person_list,bins= 30, density=True)
```

以下のシミュレーション問題に答えてください.

trial =1000000
count =0
[以下のインデントされた動作をtrial回繰り返す.]
    [x, y に0~1の間でランダムな数をそれぞれ代入する.]
    [(x, y)を座標と見たとき, もし原点(0, 0)を中心とする半径1の円の中に入っていたら(円の線上は含まない)]
        [countを1 増やす]

[ans =count/trialを求める.]
[ans を4倍した値を新しくansに代入する.]
[ansを表示する.]