

Inleiding programmeren

Martijn Stegeman & Ivo van Vulpen

<http://prognosis.mprog.nl>

Week 6: simulaties

programma

week 1: *Python syntax*

week 3: *Python syntax*

week 5: *Python syntax*

week 2

basis wiskunde

week 4

integreren + fit

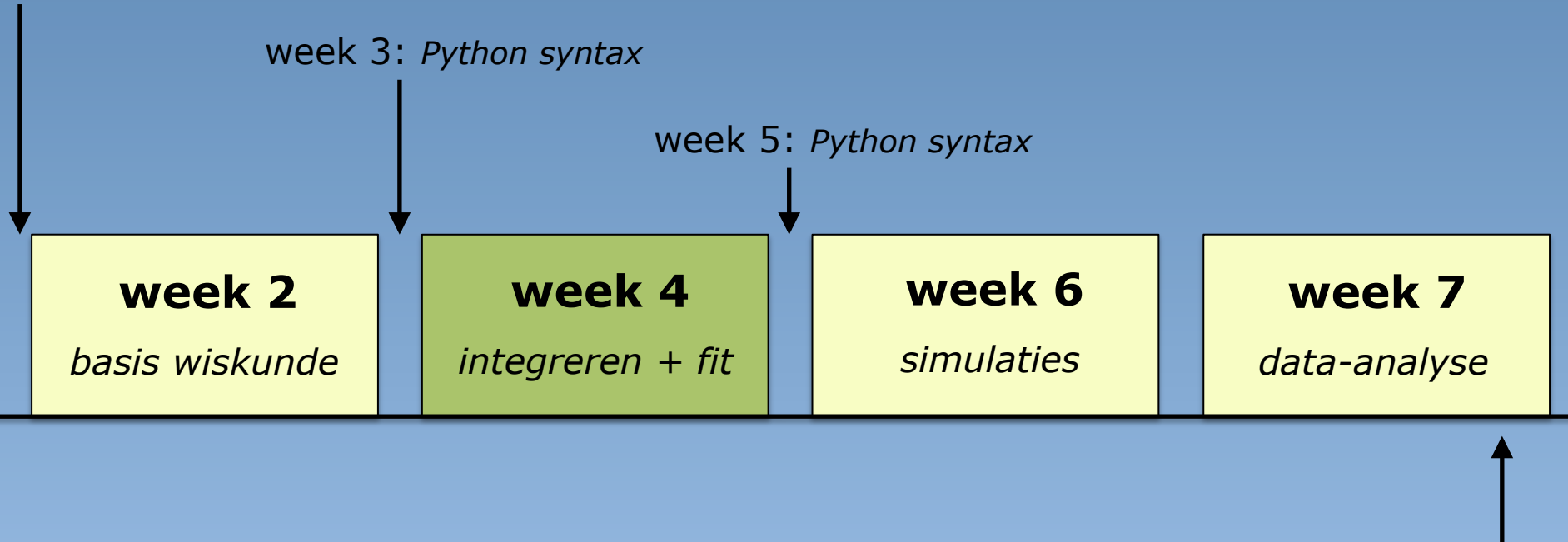
week 6


simulaties

week 7

data-analyse

Tentamen



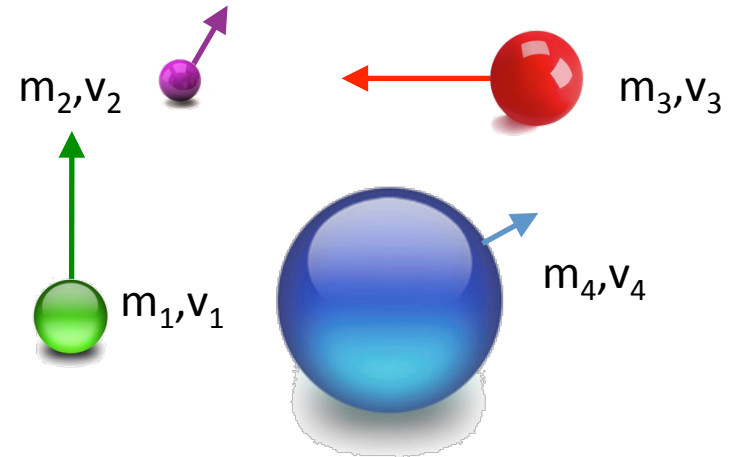


Python
bouwstenen
voor deze week

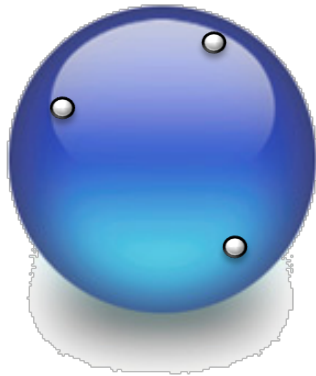
Python week 1-5

- Logica
[loops, if, random, ...]
- Lijsten, Tuples
[updaten]
- Functies
- Plotten Matplotlib

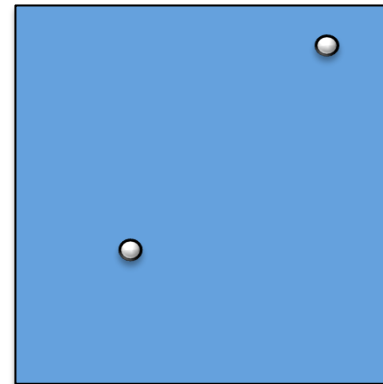
1) N-body probleem (gravitatie)



2) Geometrie



Punten op een bol equi-distant
3 / 10 ? Waar en afstanden ?

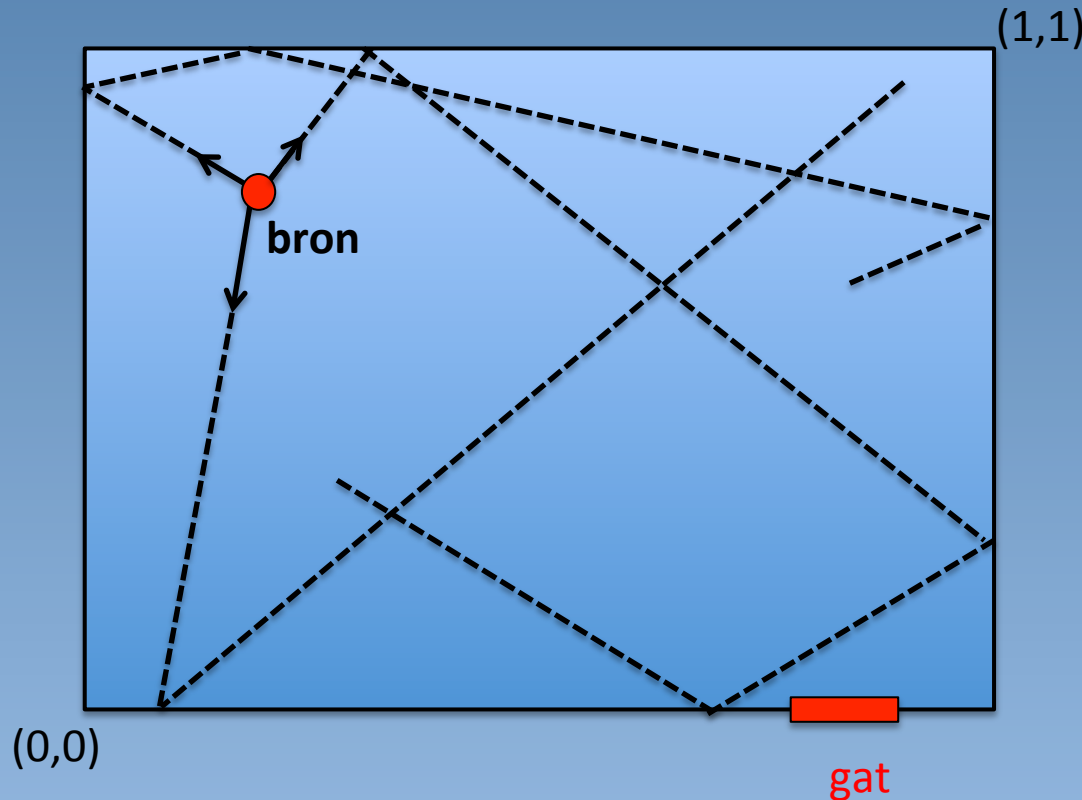


Gemiddelde afstand tussen 2 punten in
een vierkant ?

opgaves week 6

Gedrag van een groep deeltjes in de tijd

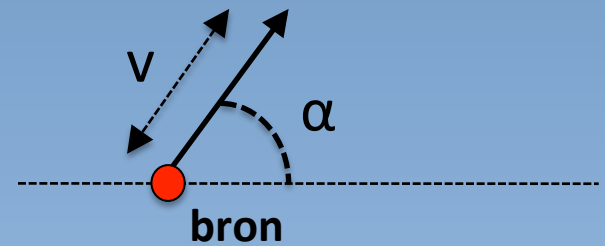
Start: in een doos ($0 < x < 1$ en $0 < y < 1$) worden op tijdstip $t=0$ vanuit een bron $(x_{\text{bron}}, y_{\text{bron}}) = (0.25, 0.75)$ een aantal deeltjes gegenereerd die elk een random snelheid en richting hebben.



Deeltjes hebben random
snelheid en richting

snelheid (v_i): $0 < v_i < 0.10$

hoek (α): $0 < \alpha < 2\pi$



Doel: kijk hoe het systeem evolueert

neem kleine stapjes in de tijd en hou voor elk deeltje

de positie x , y en de snelheid v_x en v_y bij. Tuples als vorige week (of lists)

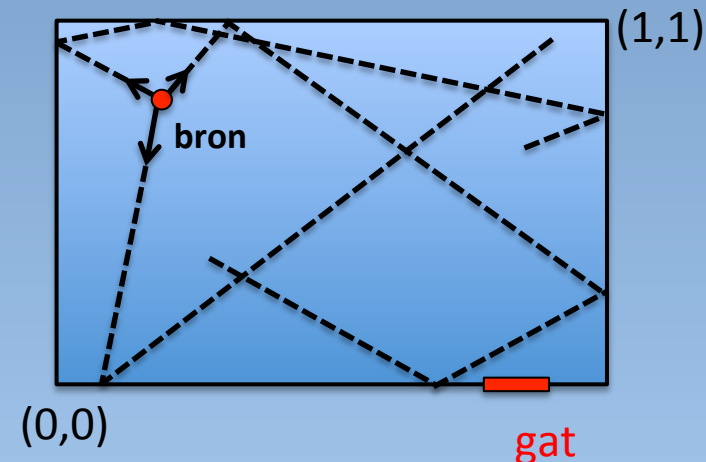
basis-opgave

Aannames:

- de deeltjes botsen elastisch tegen de wanden
- de deeltjes hebben geen afmeting en botsen niet
- in 2^e deel van de opgave zit er een gat in de doos

Opgaves:

- 1) Aantal deeltjes aan de rechter kant van de doos ($x > 0.5$) als functie van de tijd.
 - 2) Gemiddelde afstand tussen de deeltjes als functie van de tijd
- Er ontstaat nu een gat in de doos ($y_{\text{gat}} = 0$ en $0.8 < x_{\text{gat}} < 0.9$)
- 3) Aantal deeltjes in de doos als functie van de tijd. Hoe lang duurt het voor de helft van de deeltjes verdwenen is ($t_{1/2}$) ?
 - 4) Beginsnelheden 2x zo groot → wat gebeurt er met $t_{1/2}$?



Tips:

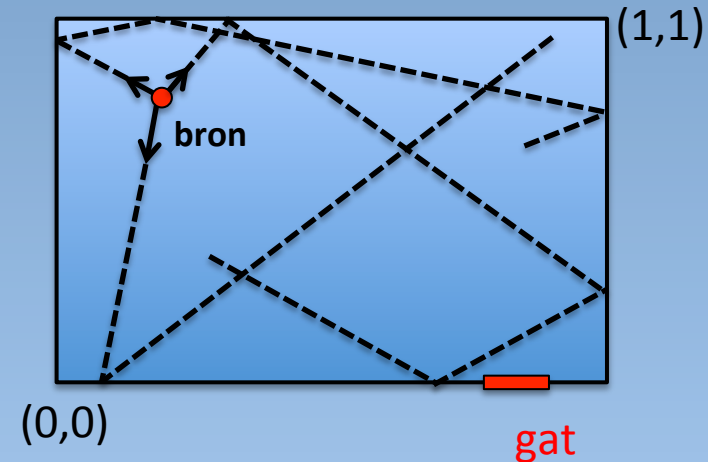
Tip 1: Neem kleine stapjes Δt in de tijd en kijk wat er verandert in positie en snelheid

$$x(t+1) = x(t) + v_x(t) * \Delta t$$

Tip 2: Hou op elk tijdstip alleen de posities (x,y) en de snelheden (v_x, v_y) van alle deeltjes bij. Transformeer hoek en absolute snelheid aan het begin dan ook gelijk in (v_x, v_y)

Tip 3: Behandel x en y afzonderlijk. Waarom kan dat in dit geval ?

Tip 4: Probeer vooraf te beredeneren wat je verwacht



Hacker

Toevoegen realisme:

- deeltjes hebben een afmeting en kunnen botsen
- animatie (film) van de evolutie van het systeem

Hacker uitbreidingen:

- 1) Gebruik animaties (template.py gegeven op de website)
- 2) Geef deeltjes een afmeting
- 3) Laat deeltjes 'echt' botsen, waarbij afmeting (= massa) van deeltjes apart kunnen worden ingesteld en de botsing ook realistisch is.
- 4) Hacker hacker: Boltzman snelheidsverdeling (vraag de assistent)

Voorbeelden: <http://www.nikhef.nl/~ivov/Animation>

