Numerieke Natuurkunde 1
Numerieke Opdrachten

Contact: Ronald Bruijn
R.Bruijn@uva.nl
Assistenten 2015:
M. Jongen
K. Melis
T. Michael
G. Sabato
T. van de Velde

versie 12
Cursus januari 2015
Gebaseerd op versie 10 van M. Vreeswijk en I. van Vulpen en
'Numerieke Natuurkunde 1: Monte Carlo Method' van K. Kroeninger
Deze syllabus is verkrijgbaar via `http://numnat.mprog.nl`

Contents

1 Inleiding **5**

1.1 begripsbepaling/afbakening . . . 5
1.2 literatuur : een keuze . . . 6
1.3 onze filosofie . . . 6
1.4 overzicht van onderwerpen . . . 6
1.5 Voorkennis . . . 7
1.6 Weergeven resultaten . . . 7
1.7 Toetsing . . . 8
1.7.1 Inleveren opdrachten . . . 9
1.8 Overige informatie . . . 9

2 Basistechnieken **10**

2.1 numeriek differentieren . . . 10
2.2 numeriek integreren . . . 12
2.3 oplossen van niet-lineaire vergelijkingen . . . 13

3 Herhaling: priemgetallen **17**

3.1 Opdracht 1 . . . 17
3.2 Opdracht 2: . . . 17
3.3 in te leveren . . . 18

4 Gewone Differentiaal Vergelijkingen (ODE's) 1e orde **19**

4.1 probleemstelling . . . 19
4.2 numerieke methoden . . . 19
4.3 opgave : radio-actief verval . . . 21
4.3.1 de fysische achtergrond . . . 21
4.3.2 de formules . . . 21
4.3.3 de procedure/de opdracht . . . 22
4.3.4 in te leveren . . . 22

5 Gewone Differentiaal Vergelijkingen (ODE's) 2e orde ; 'initial value type' **23**

5.1 probleemstelling . . . 23
5.2 numerieke methoden . . . 23
5.3 opgave : beweging deeltje bij potentiaal (klassiek) . . . 24
5.3.1 de fysische achtergrond . . . 24
5.3.2 de formules . . . 24
5.3.3 de opdracht . . . 25
5.3.4 in te leveren . . . 25

6 Het vibratie-spectrum van het waterstof molecuul **26**
6.1 physische probleembeschrijving . . . 26
6.1.1 opbouw van het programma: een advies . . . 28
6.1.2 de opdracht . . . 29
6.2 nog tijd over? vervolg onderdeel van de opdracht . . . 29

7 Partiele Differentiaal Vergelijkingen (PDE): elliptisch **31**
7.1 inleiding . . . 31
7.2 probleemstelling . . . 31
7.3 numerieke methode : in 1 dimensie, leesstof . . . 32
7.4 numerieke methode in 2 dimensies . . . 35
7.5 de opgave : electrostatische problemen . . . 37
7.5.1 de fysische achtergrond . . . 37
7.5.2 de analytische oplossing: formules . . . 37
7.5.3 de analytische oplossing: kant en klaar . . . 38
7.5.4 de analytische oplossing: de aanpak voor liefhebbers (leesstof, niet verplicht) . . . 38
7.5.5 De opdracht: de numerieke oplossing . . . 38
7.5.6 in te leveren . . . 39
7.5.7 Appendix: twee-dimensionale arrays in Python . . . 39

8 Monte-Carlo Methoden **40**
8.1 Hit or Miss . . . 40
8.2 Opdracht 1: Werpen van toevalsgetallen . . . 41
8.3 Opdracht 2: numerieke integratie . . . 41
8.4 Opdracht 3: dronkenmans wandeling . . . 42
8.5 Opdracht 4: Het file-probleem . . . 42
8.6 in te leveren . . . 43

9 En nu zelf een probleem oplossen: Rutherford scattering **44**
9.1 Basisopdracht . . . 44
9.2 Uitgebreide opdracht . . . 44
9.3 in te leveren . . . 44

10 Partiele Differentiaal Vergelijkingen (PDE): parabolisch **45**
10.1 inleiding/probleemstelling . . . 45
10.2 discretisatie van ruimte en tijd . . . 46
10.3 de expliciete methode . . . 46
10.4 een impliciete methode . . . 47
10.5 de Crank-Nicholson methode . . . 48
10.6 oplossingstechniek voor Crank Nicholson . . . 48
10.7 eenvoudige opgave . . . 50
10.7.1 in te leveren . . . 51
10.8 de opgave : een radio-aktief afval probleem . . . 52

10.8.1 de fysische achtergrond 52
10.8.2 de numerieke oplossing 52
10.8.3 de opdracht 53
10.8.4 in te leveren 54

1 Inleiding

1.1 begripsbepaling/afbakening

Numerieke Natuurkunde $\Leftrightarrow$ Computational Physics.

- Twee hoofdthema's :
  - In Klassieke Mechanica, Electro-Magnetisme, Quantummechanica, Stromingsleer etc. heb je steeds te maken met **vergelijkingen**. De toestand van een systeem en zijn verdere ontwikkeling in de tijd wordt beschreven door een (differentiaal)-vergelijking en bijbehorende **randcondities**.
    Gevraagd : los de vergelijking op!
    Soms kan dat langs analytische weg; vaak/meestal kan dat **niet** . Technieken nodig uit : *numerical analysis* .
  - In Statistische Fysica, Vaste-stoffysica etc. hebben we vaak te maken met **veel-deeltjes** problemen. De nadruk daarbij ligt niet zozeer op het oplossen van de onderliggende vergelijkingen, maar op het reconstrueren van 'meetbare' eigenschappen op basis van onderliggend collectief gedrag. Dit gebeurt door **computer-simulatie**.
    Wanneer de gesimuleerde processen een **kans**-element bevatten, spreken we van *Monte Carlo methoden*.
- De nadruk zal liggen op de eerste categorie problemen, waarvoor technieken nodig zijn uit : *numerical analysis*. Daarnaast zullen er dit jaar ook problemen behandeld worden waar *Monte Carlo methoden* voor gebruikt kunnen worden. De opdrachten staan in deze syllabus. De laaste opdracht (heel sectie 10) mag je vervangen door een vrije opdracht, waarvoor doorgaans twee sessies nodig zijn.
- De vrije opdrachten vereisen wat meer zelfstandigheid. Je moet echter wel bedenken dat deze opdrachten gemiddeld moeilijker zijn, maar vooral dat je deze opdrachten met weinig hulp zal moeten afronden. Het ligt ook niet vast wat het eindresultaat is; kijk maar hoe ver je komt en interpreteer (Vergelijk met theorie) het resultaat zelf. Hier staat tegenover dat er flink wat bonuspunten (maximaal 2) te verdienen zijn met deze keuzeopdrachten. Van alle opdrachten zijn er artikelen/verslagen aanwezig en beschikbaar via de docent met de relevante fysica en formules.
  1. Maxwell verdeling. (dit is de minst moeilijk opdracht) Simuleer de Maxwell-snelheidsverdeling van de molekulen in een ideaal gas.
  2. Van Allen gordels. Simuleer hoe deeltjes worden gevangen in de magnetosfeer van de aarde.
  3. Golfpakketje knalt tegen potentiaalberg. (dit is de meest moeilijke). Simuleer hoe een (quatummechanisch) golfpakketje tegen een potentiaalberg knalt mbv de Schrodingervegl.

1.2 literatuur : een keuze

- Boeken en Internet :

C.F. Gerald et al.	Applied Numerical Analysis
T.Pang	An Introduction to Computational Physics www.physics.unlv.edu/∼pang/cp.html
H.Gould et al.	An Introduction to Computer Simulation Methods www.kzoo.edu/∼sip/
S.E. Koonin et al.	Computational Physics
R.Landau et al.	Computational Physics,Problem Solving with Computers www.physics.orst.edu/∼rubin/CPbook
W.H.Press et al.	Numerical Recipes, the Art of Scientific Computing
N.J.Giordano	Computational Physics www/physics.purdue.edu/∼ng/comp-phys.html
J.L.Zachary	Introduction to Scientific Programming www.telospub.com/catalog/SCICOMPUTING/IntroSciProgMma.html

- Vaktijdschriften :
  Journal of Computational Physics
  Computer Physics Communications
  Computer in Physics

1.3 onze filosofie

- Niet alles 'afleiden'; het is geen wiskunde-college.
  Wel : 'geef gevoel voor'.
- Behandel slechts een klein deel van de onderwerpen;
  per onderwerp :slechts een klein deel van de technieken.
- Voorbeelden/opdrachten die aansluiten bij andere colleges
- Aandacht voor 'kwaliteit' programmatuur

1.4 overzicht van onderwerpen

Er is een uitgebreide reeks opdrachten beschikbaar.

- *Numerical Analysis*
  - Numerieke basistechnieken
  - Gewone differentiaal vergelijkingen van de 1e orde
  - Gewone differentiaal vergelijkingen van de 2e orde, type 'initial value'
  - Gewone differentiaal vergelijkingen van de 2e orde, type 'boundary value'
  - Partiele differentiaal vergelijkingen, type 'elliptisch'

  - Partiele differentiaal vergelijkingen, type 'parabolisch'

- *Monte Carlo Methoden*
  - Toevalsgetallen
  - Monte-Carlo integratie
  - Dronkemans wandeling (Random Walk)
  - 'File-probleem'

1.5 Voorkennis

In deze cursus wordt voorkennis aangenomen van de programmeertaal **Python**, zoals behandeld in het vak *Inleiding programmeren voor Natuur- en sterrenkunde (Introduction to Programming)*
In het bijzonder is het nuttig om kennis over de volgende elementen paraat te hebben:

- Het datatype **list** is onontbeerlijk om geïndexeerde lijsten van data op te slaan. Hierbij kan de index refereren naar een ruimte of tijd coordinaat, een element uit een verzameling etc. Kijk bij het gebruik van lists of andere datastructuren dat je ze met de juiste grootte initialiseert en vervolgens niet de maximale index overschrijdt bij het toekennen van een waarde. Het is uiteraard mogelijk om complexere datastructuren te gebruiken, eventueel uit een van de vele python modules

- De **for loop** is handig om bepaalde bewerkingen herhaaldelijk uit te voeren, waarbij eventueel de opeenvolgende elementen uit een datastructuur worden gebruikt of gemanipuleerd.

- Het gebruik van **functies** is zeer wenselijk om de programma code te organiseren.

1.6 Weergeven resultaten

Om de resultaten van de verschillende opgaven weer te geven, is het uitprinten van getallen niet de beste manier. Om de resultaten grafisch weer te geven zijn er grafische paketten beschikbaar voor python. Een van deze paketten is *matplotlib*. Documentatie van dit pakket is is te vinden op `matplotlib.org`. Om de handige verzamling plotfuncties *matplotlib.pyplot* te gebruiken moet dient deze natuurlijk geïmporteerd worden met

```
import matplotlib.pyplot as plt
```

of iets equivalents.

1.7 Toetsing

De toetsing geschiedt aan de hand van de in te leveren opdrachten. Deze opdrachten dienen individueel gemaakt te worden. De plagiaatregeling van de UvA is van toepassing. `http://www.uva.nl/plagiaat`

De maximale score kan behaald worden als alle opdrachten zijn voltooid. Opdrachten 1 tot en met 6 tellen bij elkaar voor 25% mee. Opdrachten 7 en 8 bij elkaar voor 25% en opdracht 9 en 10 elk 25%. Een voldoende is te behalen als alle opdrachten tot en met 9 voltooid zijn. Er is maximaal 1 bonuspunt te halen met extra of origineel werk. De programmacode van de opdrachten dient voltooid en werkend ingeleverd te worden samen met de grafische resultaten. Bij de beoordeling worden een aantal opdrachten (6,8,9 en 10) in detail bekeken waarbij extra gelet wordt op de kwaliteit van de programmatuur, overzichtelijkheid, commentaar en originaliteit.

Zoals boven beschreven, zijn de opdrachten in 4 sets verdeeld:

- Hoofdstuk/Opdracht 3 t/m 6
- Hoofdstuk/Opdracht 7 en 8
- Hoofdstuk/Opdracht 9
- Hoofdstuk/Opdracht 10

Voor elk van de sets zijn 4 sessies van 4 uur gerekend. De deadline voor de sets opdrachten is op de dag dat de laatste sessie behorende bij desbetreffende set plaatsvind, om 18.00 uur. De deadlines hangen dus af van de groep waarin je ingedeeld bent.

De code dient voorzien te zijn van commentaar. Een aantal tips:

- Begin je programma met je naam, studentnummer en programma naam.
- Geef een korte omschrijving van wat je programma doet.
- Licht toe wat je variabelen voorstellen en geef ze duidelijke namen.
- Geef een korte omschrijving bij elke functie en klasse. Hierbij moet je ook toelichten wat de argumenten zijn.
- Geef een korte omschrijving bij elk blok of paragraaf van je code. Bijvoorbeeld bij een for-loop waarin veel gebeurt.
- Geef niet op elke regel commentaar! Het is dus niet de bedoeling dat je bijvoorbeeld schrijft “tel 1 op bij i”.
- Geef wel commentaar bij kleine manipulaties die niet zo voor de hand liggen
- Schrijf het commentaar vooral voor jezelf! Je moet later je programma terug kunnen lezen en begrijpen.
- Schrijf het commentaar terwijl je bezig bent, zodat je niet achteraf moet uitzoeken wat je ook alweer gedaan hebt.

1.7.1 Inleveren opdrachten

De opdrachten dienen electronisch ingeleverd te worden. Naast de programma code, moeten de grafische resultaten ook worden ingeleverd. Het inleveren geschiedt via :

`http://numnat.mprog.nl`

Om een opdracht in te kunnen leveren, moet je ingelogd zijn met je UvA-id. Dit kan rechtsboven via *sign in*. Via het menu-item *Opdrachten* vind je de formulieren corresponderend met de verschillende hoofstukken in deze tekst.
De programma-code die ingeleverd wordt, dient direct uitvoerbaar te zijn en de grafische weergaven in een gangbaar formaat (.pdf, .ps, .eps, .jpg, .png, .tif). Meerdere grafische resultaten kunnen eventueel bij elkaar in een bestand gestopt worden met (g)zip of tar.

1.8 Overige informatie

Deze syllabus is te vinden op `numnat.mprog.nl`. Op deze site is ook hulpmateriaal en andere informatie aangaande deze cursus te vinden.

2 Basistechnieken

- Een aantal van deze technieken zijn al aan de orde gekomen bij *Inleiding programmeren voor Natuur- en sterrenkunde* . In dit hoofdstuk worden ze nog eens overzichtelijk weergegeven. Ook zijn er wat nieuwe technieken bij die U later nodig zult hebben.

- Dit hoofdstuk bevat *geen nieuwe opgaven* : U heeft er al wat aan gedaan bij vak *Inleiding programmeren voor Natuur- en sterrenkunde*

2.1 numeriek differentieren

- Probleemstelling (funktie van 1 variabele).
  Gegeven $f(x)$ op $(a \leq x \leq b)$
  Bereken $f^{'}(x)$ op $(a \leq x \leq b)$

- Verdeel interval (a, b) in N intervallen van gelijke lengte h .
  Coordinaten van de intervalgrenzen (**roosterpunten**) zijn :

$$x_i = a + ih \quad (i = 0, 1, 2.....n)$$

- Representeer $f(x)$ in de vorm van een **tabel** :

$$f_i = f(x_i) \quad (i = 0, 1, 2, ..., n)$$

- Gebruik de Taylor-reeks-ontwikkeling :

$$\boxed{f(x_i + h) = f(x_i) + hf^{'}(x_i) + \frac{h^2}{2!}f^{''}(x_i) + \mathcal{O}(h^3)} \tag{2.1}$$

  In onze notatie :

$$\boxed{f_{i+1} = f_i + hf_i^{'} + \frac{h^2}{2!}f_i^{''} + \mathcal{O}(h^3)} \tag{2.2}$$

- Voorwaartse 2-punts benadering :

$$\boxed{f_i^{'} \simeq \frac{f_{i+1} - f_i}{h} + \mathcal{O}(h)} \tag{2.3}$$

- Achterwaartse 2-punts benadering :

$$\boxed{f_i^{'} \simeq \frac{f_i - f_{i-1}}{h} + \mathcal{O}(h)} \tag{2.4}$$

- 3-punts benadering :

$$\boxed{f_i^{'} \simeq \frac{f_{i+1} - f_{i-1}}{2h} + \mathcal{O}(h^2)} \tag{2.5}$$

- Benadering 2de afgeleide :

$$\boxed{f_i'' \simeq \frac{f_{i+1} - 2f_i + f_{i-1}}{h^2} + \mathcal{O}(h^2)} \tag{2.6}$$

- Merk op :
  - De **precisie** van de benadering neemt toe naarmate h afneemt. De 'prijs' die je daarvoor betaalt is extra rekentijd ten gevolge van de toename van N
  - De **precisie** is (mede) afhankelijk van het gebruikte recept.
  - Overigens is er ook nog sprake van eem 'computer'-nauwkeurigheid.

2.2 numeriek integreren

- Probleemstelling (funktie van 1 variabele).
  Gegeven $f(x)$ op $(a \leq x \leq b)$
  Bereken
  $$\int_a^b f(x)dx$$

- Verdeel interval (a, b) in N intervallen van gelijke lengte h en representeer $f(x)$ in de vorm van een **tabel** :
  $$f_i = f(x_i) \quad x_i = a + ih \quad (i = 0, 1, 2, ..., N)$$

- Schrijf de integraal als :
  $$\int_a^b f(x)dx = \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} f(x)dx$$

- Trapezium-regel (lineaire benadering) :
  Benader $f(x)$ op het interval (x_i, x_{i+1}) door :
  $$f(x) = \frac{f_{i+1} + f_i}{2} + \mathcal{O}(h)$$
  dan volgt :
  $$\boxed{\int_a^b f(x)dx \simeq \frac{h}{2}(f_0 + 2f_1 + 2f_2..... + 2f_{N-1} + f_N) + \mathcal{O}(h^2)} \tag{2.7}$$

- Simpson-regel (parabolische benadering, N even):
  Benader $f(x)$ op het interval (x_{i-1}, x_{i+1}) door een parabool door (f_{i-1}, f_i, f_{i+1})
  $$f(x) = f_{i-1} + (x - x_{i-1})\frac{f_i - f_{i-1}}{h} +$$
  $$+\frac{(x - x_{i-1})(x - x_{i-1} - h)}{2}\frac{(f_{i+1} - 2f_i + f_{i-1})}{h^2} + \mathcal{O}(h^3)$$
  dan volgt :
  $$\boxed{\begin{aligned}\int_a^b f(x)dx = \frac{h}{3}(f_0 + f_N) + \frac{4h}{3}(f_1 + f_3.... + f_{N-1}) + \\ + \frac{2h}{3}(f_2 + f_4.... + f_{N-2}) + \mathcal{O}(h^4)\end{aligned}} \tag{2.8}$$

- Merk (weer) op :
  - De **precisie** van de benadering neemt toe naarmate h afneemt. De 'prijs' die je daarvoor betaalt is extra rekentijd ten gevolge van de toename van N
  - De **precisie** is (mede) afhankelijk van het gebruikte recept.

2.3 oplossen van niet-lineaire vergelijkingen

- Probleemstelling (funktie van 1 variabele).
  Gegeven $f(x)$ op $(a \leq x \leq b)$
  Bereken de waarde van x op het interval (a, b) waarvoor $f(x) = 0$

- we behandelen $enkele$ (eenvoudige) methoden, die we bij de verdere opgaven nodig zullen hebben. deze methoden vereisen dat de funktie $monotoon$ is op het interval (a, b).

- **de bisectie methode**.
  deze methode veronderstelt dat de funktiewaarden op de uiteinden van het $zoek\text{-}interval$ (a, b) een $tegengesteld\ teken$ hebben. we weten dan dat het nulpunt moet liggen $tussen$ $(x_l = a)$ en $(x_r = b)$. de methode gaat dan als volgt :

  - bereken de funktiewaarde op

    $$x_m = \frac{x_l + x_r}{2}$$

  - wanneer de funktiewaarde in x_m hetzelfde teken heeft als in x_l

    $$vervang \quad x_l \quad door \quad x_m$$

    wanneer de funktiewaarde in x_m hetzelfde teken heeft als in x_r

    $$vervang \quad x_r \quad door \quad x_m$$

  - ga hiermee door $totdat$ de $lengte$ van het interval (x_l, x_r) zodanig klein wordt, dat de positie van het nulpunt $voldoende\ nauwkeurig$ vastligt.
  - in het hiernavolgende $voorbeeld$ wordt , langs $iteratieve\ weg$ de wortel uitgerekend van een ingevoerd getal met de bisectie methode.

```
"""
Dit programma lost de volgende vergelijking op:
x^2-wortel^2 = 0
De wortel^2 is een gevraagde input door het programma.
"""
eps = 0.0001 # minimum intervallengte

"""
Inlezen van wortelsq
"""
wortelsqstr = raw_input("Geef wortelsq in: ")
print "We lossen op: x^2 -",wortelsqstr," = 0"
wortelsq = float(wortelsqstr)
"""
begingrenzen  zoekinterval
"""
xl = 0.
xr = wortelsq
if xr < 1. :
    xr = 1.

"""
Hier begin de while loop
"""
while (xr - xl) > eps :
    xm = (xl + xr) /2.  # midden van interval
    fl = xl*xl-wortelsq # de vgl. in xl
    fm = xm*xm-wortelsq # de vgl. in xm

    if fl*fm <= 0.:     # indien teken wisselt zijn we voorbij de oplossing
        xr = xm         # rechtergrens omlaag
    else :
        xl = xm         # linkergrens omhoog

    print "xl xr", xl, xr

"""
eindresultaat
"""
wortel = (xr+xl)/2.
onzekerheid = (xl-xr)/2.

print " wortel = ", wortel," onzekerheid = ", onzekerheid
```

- **de secant methode**
  ook deze methode veronderstelt dat de funktiewaarden op de uiteinden van het $zoek\text{-}interval\ (a, b)$ een $tegengesteld\ teken$ hebben.

$$f_l = f(a) \qquad f_r = f(b) \qquad f_l \star f_r < 0$$

  de methode gaat dan als volgt :

  - construeer een rechte lijn door (x_l , f_l) en (x_r , f_r) , en bereken het snijpunt met de x-as
$$x_s = x_l - f_l \frac{x_r - x_l}{f_r - f_l}$$
  - bereken de funktiewaarde f_s in x_s
  - wanneer de funktiewaarde in x_s hetzelfde teken heeft als in x_l

$$vervang \quad x_l \quad door \quad x_s \quad en \quad vervang \quad f_l \quad door \quad f_s$$

    wanneer de funktiewaarde in x_s hetzelfde teken heeft als in x_r

$$vervang \quad x_r \quad door \quad x_s \quad en \quad vervang \quad f_r \quad door \quad f_s$$

  - om dit iteratieve proces te $beeindigen$ zijn er twee mogelijkheden :
    * $Stop$ zodra de absolute waarde van $f(x_s)$ kleiner wordt dan 'een of andere' $afsnij\text{-}waarde$ (dicht bij 0).
    * $Stop$ zodra x_s (bijna) niet meer $verandert$.

- **methode vanuit 1 startpunt**
  soms is de situatie zodanig, dat er voor een funktie $geen\ start\text{-}interval$ ter beschikking is, maar slechts $1\ startpunt$. het is dan niet mogelijk om direct de bisectie- of de secant- methode toe te passen. we gaan dan als volgt te werk :

  - noem het $startpunt\ x_0$
  - kies een $stapgrootte\ h$, bijv. $h = 0.01\ x_0$
  - bereken de funktiewaarde f_0 in het startpunt x_0
  - bereken de funktiewaarde f_1 in $(x_1 = x_0 + h)$
  - wanneer $f_1 \star f_0 < 0$ is, hebben we het startinterval te pakken.
  - wanneer $f_1 \star f_0 > 0$ is, vergelijken we $|f_1|$ met $|f_0|$.
    als $|f_1| < |f_0|$ dan 'lopen' we kennelijk in de goede richting : ga door met 'stapjes' h, totdat het startinterval is gevonden.
    als $|f_1| > |f_0|$ dan 'lopen' we kennelijk in de verkeerde richting : keer het teken van h om, en ga door met 'stapjes' h, totdat het startinterval is gevonden.
  - zodra het startinterval is gevonden, kunnen we overschakelen op de bisectie- of de secant-methode.

- **opmerkingen** : het is duidelijk dat $alle$ beschreven methoden slechts 'werken' wanneer ze worden 'voorzien' van bruikbare $start$- informatie over de funktie waarvan het nulpunt moet worden bepaald. een $analyse\ vooraf$ van de grootheden die in een bepaald probleem een rol spelen, is hiervoor noodzakelijk.

3 Herhaling: priemgetallen

Deze eerste opdracht dient ter verfrissing van de kennis van de programmeertaal python en van de werkomgeving. De opdracht is rechtstreeks overtenomen uit de python cursus, maar dient wel weer gedaan te worden.

3.1 Opdracht 1

De opdracht luidt: Schrijf een programma $primes.py$ dat het duizendste priemgetal berekent en print. Print ook de lijst van alle 1000 priemgetallen.
Computing hints:

- een manier om te testen of een getal a een veelvoud is van getal b (b deelt a met rest 0) is het gebruik van de %-operator. In python code geeft $a\%b$ de rest ($8\%3$ is 2). Controleer de werking op de command-line.
- Gebruik list om reeksen getallen te bewaren. Bekijk de documentatie.

Hoewel een computer je in staat stel om snel te rekenen is het toch belangrijk om voor elk probleem de optimale strategie te bepalen. Hier bijvoorbeeld:

- Behalve 2 zijn even getallen nooit een priemgetal
- Verzin hoe je per priem-kandidaat bijhoudt of het wel/niet priem is als je over kandidaat delers heenloopt. Bedenkt van tevoren hoe je de lijst met gevonden priemgetallen gaat opslaan.
- Wanneer kan je stoppen ? Als je wilt bepalen of 37 een priemgetal is, welke kandidaat delers bekijk je voordat je zeker weet dat het een priemgetal is ?
- Print voor elke kandidaat informatie zodat je weet waar je bent in de berekening en je zit of de computer ook echt jouw strategie volgt.
- Zorg dat het programma stopt bij het 1000ste priemgetal. Bedenk dat je programma waarschijnlijk niet het eerste priemgetal genereerd (2).

Als je wilt controleren of je programma goed werkt, kan je je gevonden lijst priemgetallen hier matchen met een lijst bekende priemgetallen:

`http://primes.utm.edu/lists/small/1000.txt`

3.2 Opdracht 2:

Welke getallen (onder $n = 1000$) vormen de langste reeks aaneengesloten niet-priemgetallen ? Start met de code uit vraag 1.

3.3 in te leveren

Lever in:

- *primes.py* dat de resultaten van beide opdrachten produceert en op het scherm duidelijk weergeeft. Dat betekent dat er bij de afgedrukte getallen een toelichting hoort.

N.B.: : De ingeleverde programmas dienen (zoals bij alle opdrachten) te werken en de correcte resultaten te produceren.

4 Gewone Differentiaal Vergelijkingen (ODE's) 1e orde

4.1 probleemstelling

- Probleemstelling:

Onafhankelijke variabele	x op $(a \leq x \leq b)$
Afhankelijke variabele	$y(x)$
Vergelijking	$y^{'} = f(x, y)$
Randconditie	$y(a)$ gegeven
Gevraagd	$y(x)$ op $(a \leq x \leq b)$

- Voorbeeld : radioactief verval van atoomkern ; het aantal deeltjes dat per tijdseenheid vervalt, wordt gegeven door :

$$\boxed{\frac{d}{dt}N(t) = -\lambda N(t)} \tag{4.1}$$

waarin λ de levensduur van de betreffende kern is.

- Aanpak :
Verdeel interval (a, b) in N intervallen van gelijke lengte h en representeer $y(x)$ en $f(x, y)$ in de vorm van **tabellen** :

$$y_i = y(x_i) \qquad x_i = a + ih \quad (i = 0, 1, 2, ..., n)$$

$$f_i = f(x_i, y_i) \qquad (i = 0, 1, 2, ..., n)$$

y_0 is gegeven door de randconditie.
Te bepalen : $y_i \quad (i = 1, 2, ..., n)$

4.2 numerieke methoden

- Gebruik weer de Taylor-reeks-ontwikkeling :

$$\boxed{y(x_i + h) = y(x_i) + hy^{'}(x_i) + \frac{h^2}{2!}y^{''}(x_i) + \mathcal{O}(h^3)} \tag{4.2}$$

In onze notatie :

$$\boxed{y_{i+1} = y_i + hy_i^{'} + \frac{h^2}{2!}y_i^{''} + \mathcal{O}(h^3)} \tag{4.3}$$

- Methode van Euler :

$$y_i^{'} = \frac{y_{i+1} - y_i}{h} + \mathcal{O}(h) = f_i$$

$$\boxed{\Longrightarrow y_{i+1} = y_i + hf_i + \mathcal{O}(h^2)} \tag{4.4}$$

- Methode van Adams-Bashforth :

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} y_i' dx = y_i + \int_{x_i}^{x_{i+1}} f(x,y) dx$$

Benader $f(x,y)$ via lineaire extrapolatie door (x_{i-1}, y_{i-1}) en (x_i, y_i) :

$$f(x,y) \simeq \frac{x - x_{i-1}}{h} f_i - \frac{x - x_i}{h} f_{i-1} + \mathcal{O}(h^2)$$

$$\boxed{\Longrightarrow y_{i+1} = y_i + h\left(\frac{3}{2} f_i - \frac{1}{2} f_{i-1}\right) + \mathcal{O}(h^3)} \tag{4.5}$$

Merk op : doordat we bij de berekening van y_{i+1} niet alleen gebruik maken van y_i , maar ook van y_{i-1}, neemt de nauwkeurigheid toe.

- Methode van Runge-Kutta, 2e orde

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x,y) dx$$

Benader de integraal

$$y_{i+1} = y_i + h f(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}) + \mathcal{O}(h^3)$$

Gebruik Euler :

$$y_{i+\frac{1}{2}} = y_i + \frac{h}{2} f(x_i, y_i) + \mathcal{O}(h^2)$$

Dan wordt het recept :

$$\boxed{\begin{aligned} k &= h f(x_i, y_i) \\ y_{i+1} &= y_i + h f(x_i + \frac{h}{2}, y_i + \frac{k}{2}) + \mathcal{O}(h^3) \end{aligned}} \tag{4.6}$$

- Methode van Runge-Kutta, 4e orde :
Betere benadering van de integraal resulteert in het volgende recept

$$\boxed{\begin{aligned} k_1 &= h f(x_i, y_i) \\ k_2 &= h f(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}) \\ k_3 &= h f(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}) \\ k_4 &= h f(x_i + h, y_i + k_3) \\ y_{i+1} &= y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) + \mathcal{O}(h^5) \end{aligned}} \tag{4.7}$$

- Pas op !! de aangegeven orde van de fout betreft de **locale** fout.
De **globale** fout is een factor h^{-1} groter.

- Merk (weer) op :
  - De **precisie** van de benadering neemt toe naarmate h afneemt.
  - De **precisie** is (mede) afhankelijk van het gebruikte recept.

4.3 opgave : radio-actief verval

4.3.1 de fysische achtergrond

- het radio-actief verval van een atoom wordt beschreven door

$$\boxed{\frac{d}{dt}N(t) = -\lambda N(t) \qquad met \qquad \lambda = \frac{\ln 2}{T_{\frac{1}{2}}}} \tag{4.8}$$

  waarin $T_{\frac{1}{2}}$ de halveringstijd is van het vervallende atoom.

- volgens de 'handboeken' bestaat er het volgende verval :

$$\boxed{{}^{76}Br \quad \rightarrow \quad {}^{76}Se} \tag{4.9}$$

  de halveringstijd van Bromine is 16.1 uur.
  het eindprodukt Selenium is stabiel.

- we gaan uit van een beginsituatie waarin alleen Bromine aanwezig is.
  de **opdracht** is om uit te rekenen hoeveel Bromine en Selenium er aanwezig is, als functie van de tijd.

4.3.2 de formules

- noem de beginhoeveelheid Bromine : N_0

- noem de hoeveelheden Bromine en Selenium als functie van de tijd : $N_1(t), N_2(t)$.
  noem de vervalsconstante van Bromine : λ_1

- de hoeveelheid Bromine, als functie van de tijd, wordt gegeven door :

$$\boxed{\frac{d}{dt}N_1(t) = -\lambda_1 N_1(t) \quad met \;\; N_1(t=0) = N_0} \tag{4.10}$$

- de hoeveelheid Selenium wordt verkregen door 'sommatie' van het Bromine verval.

- de vergelijkingen voor N_1 heeft een analytische oplossing ;

$$\boxed{N_1(t) = N_0 \exp(-\lambda_1 t)} \tag{4.11}$$

- **Merk op** dat de differentiaalvergelijking bij deze opgave de vorm heeft van $y' = f(y)$. in het 'recept' voor de Runge-Kutta methodes vervalt derhalve de afhankelijkheid van x.

4.3.3 de procedure/de opdracht

- bij de numerieke berekening nemen we steeds kleine stapjes in de tijd, waarbij we de hoeveelheid vervallend Bromine berekenen.
  - gebruik als tijdstap $(0.01)*$ de halveringstijd van Bromine.
  - neem 750 tijdstappen.
- dit resultaat wordt gebruikt om de 'resterende' hoeveelheid Bromine te berekenen, evenals de hoeveelheid geproduceerd Selenium.
- gebruik de analytische oplossing om de precisie van de numerieke procedure te controleren. doe dit door op elk tijdstip de absolute waarde uit te rekenen van het verschil tussen de numerieke en de analytische waarde, waarbij dit verschil wordt gedeeld door de analytische waarde.
- gebruik eerst de methode van Euler, en dan de methode van Runge-Kutta (4e orde)
- om het resultaat van de berekeningen zichtbaar te maken, is het 'uitprinten' van getallen niet erg geschikt : gebruik een **grafisch pakket** .

4.3.4 in te leveren

Lever in :

- *uw programma-code (decay.py)*
- *de grafische resultaten van* :
  - *de hoeveelheden Bromine en Selenium als functie van de tijd*
  - *de precisie van de hoeveelheid Bromine, voor de Euler methode, en voor de Runge-Kutta methode*

5 Gewone Differentiaal Vergelijkingen (ODE's) 2e orde ; 'initial value type'

5.1 probleemstelling

- Probleemstelling:

Onafhankelijke variabele	x op $(a \leq x \leq b)$
Afhankelijke variabele	$y(x)$
Vergelijking	$y^{''} = f(x, y, y^{'})$
Randcondities	$y(a) = y_0$ en $y^{'}(a) = y_0^{'}$ gegeven.
Gevraagd	$y(x)$ op $(a \leq x \leq b)$

- Voorbeeld : de vergelijking van Newton uit de Klassieke Mechanica

$$\boxed{\frac{d^2y(t)}{dt^2} = \frac{F(y)}{m}} \tag{5.1}$$

als de plaats $y(t)$ en de snelheid $y^{'}(t)$ op het tijdstip $t = 0$ gegeven zijn, en natuurlijk ook de massa m en de kracht F , dan ligt het verdere verloop van de plaats en de snelheid vast, en kan worden berekend.

- wanneer de randcondities op deze wijze zijn gegeven (beide in $hetzelfde$ punt), spreken we van '$initial\ value\ type$'.

5.2 numerieke methoden

- vergelijkingen van deze vorm, en met deze randcondities, kunnen worden **herleid** tot twee gewone differentiaal- vergelijkingen van de 1e orde, met ieder hun eigen randconditie.

- introduceer de snelheid als 'hulp'variabele. de vergelijkingen worden dan :

$$\boxed{\frac{dy(t)}{dt} = v(t) \qquad \frac{dv(t)}{dt} = \frac{F(y)}{m}} \tag{5.2}$$

met $y(t = 0)$ en $v(t = 0)$ gegeven door de randcondities.

- de numerieke aanpak wordt nu als volgt :
  - introduceer een tijdstap δt, en discretiseer tijd, plaats, snelheid en kracht :

$$t_i = t_0 + i\delta t \quad y_i = y(t_i) \quad v_i = v(t_i) \quad F_i = F(y_i) \quad (i = 0, 1, 2, ..)$$

  - de vergelijkingen worden dan :

$$\boxed{\frac{dy_i}{dt} = v_i \qquad \frac{dv_i}{dt} = \frac{F_i}{m}} \tag{5.3}$$

met y_0 en v_0 gegeven door de randcondities.

- de rekenprocedure wordt dan :
  - bereken y_1 uit y_0 , gebruikmakend van v_0
  - bereken v_1 uit v_0 , gebruikmakend van F_0 en m.
  - bereken y_2 uit y_1 , gebruikmakend van v_1.
  - etc.
- merk op dat deze aanpak slechts 'werkt' omdat de randcondities van plaats en snelheid **op hetzelfde tijdstip** gegeven zijn.

5.3 opgave : beweging deeltje bij potentiaal (klassiek)

5.3.1 de fysische achtergrond

- we nemen een probleem uit de Klassieke Mechanica : de 1-dimensionale beweging van een deeltje in aanwezigheid van een **potentiaalberg** $V(x)$
- de vergelijking van Newton heeft dan de vorm :

$$\boxed{\frac{d^2x(t)}{dt^2} = -\frac{1}{m}\frac{dV(x)}{dx}} \tag{5.4}$$

  waarbij m de massa van het deeltje is.

- we geven de potentiaalberg de vorm van een 'Gaussische verdeling' rond het punt $x = x_{pot}$, met een $breedte = 1$, en een maximale hoogte V_{max} :

$$\boxed{V(x) = V_{max} \exp\left[-\frac{(x - x_{pot})^2}{2}\right]} \tag{5.5}$$

- we starten op het tijdstip $t = 0$ met een deeltje met een gegeven beginsnelheid $v = v_0$, en op een plaats $x = x_0$ waar de potentiaal $V(x_0) \simeq 0$.
  we laten het deeltje lopen in de richting van de potentiaalberg, en berekenen het verloop van plaats en snelheid als funktie van de tijd.
- voor het gemak kiezen we de massa van het deeltje ($m = 1$). daardoor wordt de snelheid gelijk aan de impuls.

5.3.2 de formules

- we kunnen de (2e orde) vergelijking van Newton schrijven als een combinatie van twee (1e orde) vergelijkingen :

$$\boxed{\begin{aligned} \frac{dx(t)}{dt} &= p(t) \\ \frac{dp(t)}{dt} &= -\frac{dV(x)}{dx} = V_{max}\,(x - x_{pot}) \exp\left[-\frac{(x - x_{pot})^2}{2}\right] \end{aligned}} \tag{5.6}$$

- na discretisatie van tijd, plaats, en snelheid/impuls worden de vergelijkingen

$$\boxed{\begin{aligned}\frac{dx_i}{dt} =& p_i \\ \frac{dp_i}{dt} =& -\frac{dV(x=x_i)}{dx} = V_{max}\left(x_i - x_{pot}\right)\exp\left[-\frac{(x_i - x_{pot})^2}{2}\right]\end{aligned}} \tag{5.7}$$

met x_0 en p_0 gegeven.

- een praktische opmerking : opdat het programma de potentiaalberg *goed ziet* , moet de *tijdstap* in combinatie met de beginimpuls zodanig worden gekozen, dat de *verplaatsing* van het deeltje per tijdstap aanzienlijk kleiner is dan de breedte van de potentiaalberg.

$$\boxed{ds = pdt << 1} \tag{5.8}$$

5.3.3 de opdracht

- interessante grootheden bij dit probleem zijn niet allen de ontwikkeling van plaats en impuls van het deeltje, als funktie van de tijd, maar ook de ontwikkeling van de kinetische energie E_{kin}, de potentiele energie E_{pot} en de totale energie E_{tot} :

$$\boxed{E_{kin}(t) = \frac{p^2(t)}{2} \qquad E_{pot}(t) = V\left(x(t)\right) \qquad E_{tot}(t) = E_{kin}(t) + E_{pot}(t)} \tag{5.9}$$

- de totale energie moet **behouden** zijn, en gelijk aan de beginenergie :

$$\boxed{E_{tot}(t) = E_{tot}(t=0) = \frac{p^2(t=0)}{2} = constant} \tag{5.10}$$

- verder zijn er twee interessante situaties te ondescheiden :
  - de beginenergie is *kleiner* dan de hoogte van de potentiaalberg
  - de beginenergie is *groter* dan de hoogte van de potentiaalberg
- de **opdracht** luidt :
  - bereken voor **beide** 'soorten' beginenergieen, het verloop in de tijd van plaats, impuls en de energieen. Gebruik hierbij de methode van Adams-Bashforth.
  - gebruik een functie om de afgeleide van te potentiaal te berekenen.
  - maak het resultaat langs **grafische** weg zichtbaar.

5.3.4 in te leveren

Lever in :

- *uw programma-code (particle1dim.py)*
- *grafische resultaten.* Deze moeten minstens voor beide gevallen bevatten de plaats, impuls, en de drie energieën.

6 Het vibratie-spectrum van het waterstof molecuul

6.1 physische probleembeschrijving

- We beschouwen een molecuul, bestaand uit twee gelijksoortige atomen (bijv. waterstof H_2 of zuurstof O_2).
- De potentiaal tussen beide atomen (ten gevolge van de electrische wisselwerkingen tussen de atoomkernen en de electronen) heeft de volgende componenten :
  - een **afstotende** component, op **zeer kleine** afstanden.
  - een **aantrekkende** component, op **grote** afstanden.
- een voorbeeld van een dergelijke potentiaal is afkomstig van Lennard-Jones

$$\boxed{V(r) = 4V_0 \left[\left(\frac{a}{r}\right)^{12} - \left(\frac{a}{r}\right)^{6} \right]} \tag{6.1}$$

  - $V(r) = 0$ voor $r = a$
  - op kleine afstanden ($r < a$) is $V(r)$ positief : de atomen stoten elkaar af.
  - op afstanden ($r > a$) is $V(r)$ negatief : de atomen trekken elkaar aan.
  - de minimumwaarde van de potentiaal $= -V_0$. deze wordt bereikt bij

$$\boxed{r_{min} = 2^{\frac{1}{6}} a} \tag{6.2}$$

  - op grote afstanden ($r \gg r_{min}$) gaat de (negatieve) potentiaal asymptotisch naar 0.
  - De waarden van V_0 en a hangen af van het betreffende atoom.
- De atomen kunnen ten opzichte van elkaar $vibreren$: een beweging waarbij ze zich beurtelings 'van elkaar af' en 'naar elkaar toe' bewegen.
  De beweging 'van elkaar af' wordt (op de duur) gestopt doordat de aantrekkende kracht te groot wordt.
  De beweging 'naar elkaar toe' wordt (op een gegeven moment) gestopt doordat de afstotende kracht te groot wordt.
- Tijdens de beweging geldt de behoudswet van energie-impuls :

$$\boxed{E_{totaal} = \frac{p^2(r)}{2m} + V(r) = constant} \tag{6.3}$$

  In de 'omkeerpunten' r_{in} en r_{out} geldt :

$$\boxed{E_{totaal} = V(r_{in}) = V(r_{out})} \tag{6.4}$$

- Bekend is dat voor gebonden systemen op atomaire schaal **niet alle** waarden van E_{totaal} zijn toegestaan: er is een **spectrum** van energie-**eigenwaarden**. De quantisatie-regels van Sommerfeld-Bohr komen er op neer dat alle gebonden toestanden moeten voldoen aan de volgende eis :

$$\boxed{\int_{r_{in}}^{r_{out}} p(r)dr = \sqrt{2m}\int_{r_{in}}^{r_{out}} [E_n - V(r)]^{\frac{1}{2}}\, dr = (n+\frac{1}{2})\hbar\pi} \tag{6.5}$$

  Hierin is $\hbar$ de constante van Planck (gedeeld door 2π),
  en n = 0, 1, 2,.....

- Het berekenen van de energie-eigenwaarden komt dus neer op het oplossen van de vergelijking :

$$\boxed{\sqrt{2m}\int_{r_{in}}^{r_{out}} [E_n - V(r)]^{\frac{1}{2}}\, dr = (n+\frac{1}{2})\hbar\pi} \tag{6.6}$$

  waarbij r_{in} en r_{out} de oplossingen zijn van

$$\boxed{E_n = V(r) = 4V_0\left[\left(\frac{a}{r}\right)^{12} - \left(\frac{a}{r}\right)^{6}\right]} \tag{6.7}$$

- Om het rekenwerk te vereenvoudigen, gaan we over op andere eenheden:
  - gebruik a als eenheid van lengte : $r \to ax$
  - gebruik V_0 als eenheid van energie : $E_n \to e_n V_0$
  - introduceer $\gamma = a\hbar^{-1}\sqrt{2mV_0}$

  Het berekenen van de energie-eigenwaarden komt nu neer op het oplossen van de vergelijking :

$$\boxed{\gamma\int_{x_{in}}^{x_{out}} [e_n - v(x)]^{\frac{1}{2}}\, dx = (n+\frac{1}{2})\pi} \tag{6.8}$$

  waarbij x_{in} en x_{out} de oplossingen zijn van

$$\boxed{e_n = v(x) = 4\left[x^{-12} - x^{-6}\right]} \tag{6.9}$$

  en waarbij (volgens de handboeken) voor H_2 geldt

$$\boxed{\gamma = 21.7} \tag{6.10}$$

- merk op dat in deze eenheden
  - $V(x) = 0$ voor $x = 1$
  - de minimumwaarde van de potentiaal $= -1$. deze wordt bereikt bij

$$\boxed{x_{min} = 2^{\frac{1}{6}}} \tag{6.11}$$

6.1.1 opbouw van het programma: een advies

Deze opdracht resulteert in een tamelijk 'lang' programma. Het is belangrijk de zaak stap voor stap op te zetten, en na elke stap te testen of het bijbehorende programma-deel goed werkt. Dit kan bijv. op de volgende manier :

- maak een functie **vpot (x)** die voor elke waarde van x de waarde van de Lennard-Jones potentiaal $v(x)$ levert.
  U kunt aan de formule van de potentiaal een aantal dingen $zien$, die $nuttig$ zijn om Uw functie te $testen$:
  - De potentiaal heeft een minimum waarde $v = -1$ in $x = x_{min} = 2^{\frac{1}{6}}$
  - Voor het snijpunt met de x-as geldt : $v(x = 1) = 0$
  - Voor 'kleine' waarden van x, is x^{-12} de dominerende term
  - Voor 'grote' waarden van x, is $-x^{-6}$ de dominerende term
- Om Uw functie **vpot** te $testen$, kunt U bijv. als volgt te werk gaan :
  - Bereken de waarden van de potentiaal op het interval $0.96 < x < 5.96$ met tussenstapjes van 0.01
  - Maak een grafische plot van het resultaat.
- Bij een $gekozen$ waarde van de energie e hoort een waarde voor de zgn. actie-integraal.
  $$s(e) = \gamma \int_{x_{in}}^{x_{out}} [e - v(x)]^{\frac{1}{2}}\, dx$$
  Om deze waarde te berekenen zijn nodig de beide oplossingen x_{in} en x_{out} van de vergelijking $e = v(x)$. Dus :
- Maak een functie **action (e)** , die :
  - voor een $gegeven$ waarde van e de bijbehorende waarden van x_{in} en x_{out} berekent (dit kan langs analytische weg)
  - de bijbehorende waarde van de actie-integraal berekent via numerieke integratie (bijv. met de trapezium-regel)
- Om Uw functie **action** te $testen$, kunt U bijv. als volgt te werk gaan :
  - Roep de functie aan met achtereenvolgens
    $e = -0.9, -0.8, -0.7, ... - 0.1$
    Print de waarden van x_{in} , x_{out} en van de actie-integraal als functie van e uit. Kijk of dat er uit ziet als verwacht.
  - Als er zich energie-eigenwaarden bevinden binnen het bereik van de bovenstaande e's, dan moeten we zien dat de waarden van de actie-integraal de waarden van $(n + \frac{1}{2})\pi$ omvatten. Verifieer dit.

- We hebben nu alle ingredienten die nodig zijn om de laagste energie-eigenwaarde e_0 te vinden. Daarvoor moet gelden:

$$s(e_0) - \frac{\pi}{2} = 0.$$

Dit *komt neer* op het numeriek oplossen van een (niet-lineaire) vergelijking, zoals besproken in het hoofdstuk 'Basistechnieken'. Kandidaat-methodes zijn : 'Bisectie' en 'Methode vanuit 1 startpunt'. Dus :

- Maak een funktie
eigen (estart, value) die de waarde van de energie uitrekent waarvoor $s(e) - value = 0$.
Gebruik $estart$ als 'start-waarde' of als 'ondergrens' van de zoekmethode.
Geef de gevonden eigenwaarde van de energie terug als $returnwaarde$ van de functie.

- De energie-eigenwaarde e_1 moet voldoen aan :

$$s(e_1) - \frac{3\pi}{2} = 0.$$

Deze waarde kunnen we weer vinden met de funktie **eigen**, aangeroepen met een andere waarde voor $value$. Daarbij is $estart = e_0$ een goede startwaarde .

- Voor 'hogere' energie eigenwaarden, kan dezelfde procedure worden voortgezet.

- **PS** een nauwkeurige waarde voor π krijgt U via de 'opdracht' $pi = acos\ (-1.)\ ;$

6.1.2 de opdracht

De opdracht is :

- *bereken de eigenwaarden van de energie, voorzover ze* **kleiner dan** -0.01 *zijn.*

- *lever in*

  - *uw programma-code (hspec.py)*
  - *grafische resultaten*

6.2 nog tijd over? vervolg onderdeel van de opdracht

- in de literatuur vinden we ,als experimenteel resultaat, een tabel met $twaalf$ energie-eigenwaarden $kleiner\ dan\ -0.01$.
Het zal duidelijk zijn dat de berekeningen met de Lennard-Jones potentiaal niet goed kloppen met het experimentele resultaat.

- een beter resultaat kan worden verkregen als we de Lennard-Jones potentiaal vervangen door de Morse potentiaal

$$\boxed{v(x) = \left[1 - \exp\left(-\beta(x - x_{min})\right)\right]^2 - 1} \tag{6.12}$$

met

$$\boxed{x_{min} = 2^{\frac{1}{6}} \qquad \beta = 1.5} \tag{6.13}$$

- de **opgave** luidt : herhaal het zoeken naar eigenwaarden met de Morse potentiaal, en vergelijk het numerieke resultaat met het experiment.

7 Partiele Differentiaal Vergelijkingen (PDE): elliptisch

7.1 inleiding

Wanneer de funkties die we willen uitrekenen, afhankelijk zijn van $meer\ dan\ een$ variabele, dan worden de afgeleiden $partiele$ afgeleiden; we spreken dan van $partiele$ differentiaalvergelijkingen. Dit doet zich voor wanneer de funktie het gedrag beschrijft van een systeem in een twee- of drie- dimensionale ruimte, of wanneer de funktie afhangt van zowel plaats als tijd.
De algemene vorm van een 2e orde partiele differentiaal vergelijking van een funktie van twee variabelen is :

$$\boxed{A\frac{\partial^2 u}{\partial x^2} + B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} + D\left(x, y, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right) = 0} \tag{7.1}$$

De vergelijking heet $elliptisch$, als $B^2 - 4AC < 0$
De vergelijking heet $parabolisch$, als $B^2 - 4AC = 0$
De vergelijking heet $hyperbolisch$, als $B^2 - 4AC > 0$

7.2 probleemstelling

- We beschouwen een probleem uit de $Electrostatica$: in een ruimte waarin geen electrische ladingsbronnen aanwezig zijn, moet de electrische potentiaal $V(x, y, z)$ voldoen aan de Laplace- vergelijking :

$$\boxed{\Delta V(x, y, z) = \frac{\partial^2 V(x, y, z)}{\partial x^2} + \frac{\partial^2 V(x, y, z)}{\partial y^2} + \frac{\partial^2 V(x, y, z)}{\partial z^2} = 0} \tag{7.2}$$

- Om wille van de eenvoud beperken we ons tot een $2\text{-}dimensionaal$ probleem; de vergelijking wordt dan :

$$\boxed{\Delta V(x, y) = \frac{\partial^2 V(x, y)}{\partial x^2} + \frac{\partial^2 V(x, y)}{\partial y^2} = 0} \tag{7.3}$$

- De 'ruimte' waarop $V(x, y)$ moet worden berekend is beperkt door 'randen' (in plaats van eindpunten). wij $begrenzen$ de 'ruimte' door een rechthoekig gebied :

$$x_{min} \leq x \leq x_{max} \qquad y_{min} \leq y \leq y_{max}$$

- De benodigde $randcondities$ kunnen twee vormen aannemen :
  - Dirichlet type : legt de waarde van de funktie op een rand vast
  - Von Neumann type : legt waarde van de **afgeleide** van de funktie (**loodrecht** op een rand) vast.

7.3 numerieke methode : in 1 dimensie, leesstof

De numerieke methode die wordt gebruikt om bovenstaand type vergelijking op te lossen, kan het makkelijkst worden beschreven aan de hand van een $gewone$ differentiaalvergelijking van de 2e orde, van het 'boundary value' type. Aan het 1 dimensionale probleem is overigens geen opdracht verbonden.

- we beschouwen de vergelijking :

$$\boxed{\frac{d^2\phi}{dx^2} = S(x) \quad (a \leq x \leq b)} \tag{7.4}$$

  met randcondities: $\phi(a)$ en $\phi(b)$ gegeven.

- Verdeel interval (a, b) in N intervallen van gelijke lengte h met **roosterpunten**

$$x_i = a + ih \quad (i = 0, 1, 2.....n)$$

- Representeer $\phi(x)$ in de vorm van een **tabel** :

$$\phi_i = \phi(x_i) \quad (i = 0, 1, 2, ..., n)$$

- Randcondities leggen ϕ_0 en ϕ_n vast.

- Representeer $S(x)$ in de vorm van een **tabel** :

$$S_i = S(x_i) \quad (i = 0, 1, 2, ..., n)$$

- Benader de 2e afgeleide :

$$\frac{d^2\phi_i}{dx^2} = \frac{\phi_{i-1} - 2\phi_i + \phi_{i+1}}{h^2}$$

- De vergelijking wordt dan herleid tot een stelsel van $(n-1)$ vergelijkingen

$$\boxed{\phi_{i-1} - 2\phi_i + \phi_{i+1} = h^2 S_i \quad (i = 1, 2, ..., n-1)} \tag{7.5}$$

- Je kunt dit interpreteren als een matrix-vergelijking:

$$\begin{pmatrix} -2 & 1 & 0 & 0 & ... & 0 \\ 1 & -2 & 1 & 0 & ... & 0 \\ 0 & 1 & -2 & 1 & ... & 0 \\ . & . & . & . & ... & . \\ . & . & . & . & ... & . \\ . & . & . & 1 & -2 & 1 \\ . & . & . & 0 & 1 & -2 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ ... \\ ... \\ ... \\ \phi_{n-2} \\ \phi_{n-1} \end{pmatrix} = \begin{pmatrix} h^2 S_1 - \phi_0 \\ h^2 S_2 \\ ... \\ ... \\ ... \\ h^2 S_{n-2} \\ h^2 S_{n-1} - \phi_n \end{pmatrix} \tag{7.6}$$

Oplossing van de vergelijking $A\vec{\phi} = \vec{b}$ is

$$\vec{\phi} = A^{-1}\vec{b}$$

Maar: de matrix heeft een bijzondere structuur : $\mathbf{sparse}$, $\mathbf{tridiagonal}$, $\mathbf{blokken}$. Vandaar dat andere oplossingsrecepten (anders dan matrix-inversie en vermenigvuldiging) efficienter kunnen zijn (in termen van geheugenruimte en/of rekentijd)

- We behandelen de methode van $\mathbf{Gauss\text{-}Seidel\ iteratie}$.
  Bij deze methode hanteren we een iteratieve procedure die uiteindelijk moet leiden tot een set waarden voor ϕ_i die voldoet aan vgl (7.5).
  De methode gaat als volgt :
  - Zet de waarden van ϕ_0 en ϕ_n in de eindpunten vast op de randvoorwaarden. Deze waarden mogen verder $\mathbf{niet}$ veranderd worden.
  - Zet de waarde van ϕ_i in de overige punten op een willekeurige startwaarde, bijv: $\phi_i = 0 \quad (i = 1, 2, ..., n-1)$. Duidelijk is dat deze set waarden voor ϕ_i niet aan de voorwaarden voldoet.
  - Loop vervolgens de punten $(i = 1, 2, ..., n-1)$ een voor een af, en vervang de 'oude waarde' van ϕ_i door een 'nieuwe waarde'

    $$\boxed{vervang\ \phi_i\ door\ \ \frac{1}{2}\left(\phi_{i+1} + \phi_{i-1} - h^2 S_i\right)} \tag{7.7}$$

    Vervang de 'oude waarde' $\mathbf{onmiddellijk}$ door de 'nieuwe!!
  - De nieuwe set waarden voor ϕ_i zal $\mathbf{beter}$ aan de voorwaarden voldoen, maar mogelijkerwijze nog $\mathbf{niet\ goed\ genoeg}$. Vandaar dat we de vervangingsprocedure een aantal malen herhalen. Als alles goed gaat zal het resultaat convergeren naar de juiste oplossing.

- Overgang naar 'bewijs' van convergentie.
  Beschouw de grootheid

$$\boxed{E = \int_a^b dx \left[\frac{1}{2}\left(\frac{d\phi}{dx}\right)^2 + S\phi\right]}$$

$$\boxed{= \frac{1}{2h}\sum_{i=1}^{n}\left(\phi_i - \phi_{i-1}\right)^2 + h\sum_{i=1}^{n-1} S_i\phi_i} \tag{7.8}$$

Voor de 1ste afgeleide van E naar een bepaalde ϕ_j geldt :

$$\frac{\partial E}{\partial \phi_j} = \frac{1}{h}\left(2\phi_j - \phi_{j-1} - \phi_{j+1}\right) + hS_j$$

Dus : als ϕ_{j-1} , ϕ_j en ϕ_{j+1} aan de basisvergelijking voldoen, geldt :

$$\frac{\partial E}{\partial \phi_j} = 0$$

Voor de 2e afgeleide van E naar een bepaalde ϕ_j geldt :

$$\frac{\partial^2 E}{\partial \phi_j^2} = \frac{2}{h} > 0$$

Dus : voor de 'correcte' set waarden van ϕ_i (die aan de basisvergelijking voldoet) geldt dat de grootheid E zijn **minimale** waarde heeft.

- Schets 'bewijs' van convergentie.
  In onze procedure hebben we op elk moment een set waarden

  $$\phi_i \quad (i = 0, 1, ..., n)$$

  Daarbij behoort een zekere waarde van E.
  Door onze substitutie (7.7) verandert de waarde van E met :

  $$-\frac{1}{h}\left[\frac{1}{2}(\phi_{i+1} + \phi_{i-1} - h^2 S_i) - \phi_i\right]^2 \leq 0$$

  Dus E wordt kleiner en we zijn op de (goede) weg naar het minimum.

- **Relaxatie**. We kunnen de substitutie (7.7) ook schrijven als :

  $$\boxed{vervang \quad \phi_i \quad door \quad \phi_i + \frac{1}{2}\left(\phi_{i+1} + \phi_{i-1} - 2\phi_i - h^2 S_i\right)} \tag{7.9}$$

  waarbij de 'nieuwe' waarde van ϕ_i uit de oude wordt verkregen door er een correctie-term bij op te tellen. Deze correctie-term $= 0$ als de oplossing is bereikt.

- Een iets algemener 'recept' is :

  $$\boxed{vervang \quad \phi_i \quad door \quad (1-\omega)\phi_i + \frac{\omega}{2}\left(\phi_{i+1} + \phi_{i-1} - h^2 S_i\right)} \tag{7.10}$$

  De verandering ΔE in E ten gevolge van deze substitutie is :

  $$-\frac{\omega(2-\omega)}{2h}\left[\frac{1}{2}(\phi_{i+1} + \phi_{i-1} - h^2 S_i) - \phi_i\right]^2 \leq 0$$

  - Zolang $0 < \omega < 2$ is $\Delta E \leq 0$, dus is de procedure **convergent**.
  - De keuze $\omega = 1$ komt overeen met **geen-relaxatie**.
    De ervaring heeft uitgewezen dat de convergentie vaak kan worden **versneld** door $\omega \neq 1$ te kiezen.
    Als $\omega < 1$ spreken we van **over-relaxatie**.
    Als $\omega > 1$ spreken we van **onder-relaxatie**.
  - Een **algemeen** recept voor de **beste** keuze van ω is niet te geven : probeer het uit !

7.4 numerieke methode in 2 dimensies

De bovenbeschreven methode laat zich makkelijk generaliseren tot 2 dimensies. Lees de onderstaande stof eerst goed door en dan volgt de opdracht om een numerieke oplossing te vinden die je kunt controleren met een kant en klaar programma met de analystische oplossing.

- de vergelijking voor de potentiaal was :

$$\boxed{\Delta V(x,y) = \frac{\partial^2 V(x,y)}{\partial x^2} + \frac{\partial^2 V(x,y)}{\partial y^2} = 0} \tag{7.11}$$

- De 'ruimte' waarop $V(x,y)$ moet worden berekend werd beperkt door de 'randen'

$$\boxed{x_{min} \leq x \leq x_{max} \qquad y_{min} \leq y \leq y_{max}}$$

- We nemen aan dat er Dirichlet randcondities zijn, die de potentiaal op de randen vastleggen.

- Verdeel interval (x_{min}, x_{max}) in N intervallen van gelijke lengte h . Coordinaten van de intervalgrenzen zijn :

$$x_i = x_{min} + ih \quad (i = 0, 1, 2....., n)$$

- Verdeel interval (y_{min}, y_{max}) in M intervallen van gelijke lengte h . Coordinaten van de intervalgrenzen zijn :

$$y_j = y_{min} + jh \quad (j = 0, 1, 2....., m)$$

- Roosterpunten hebben coordinaten (x_i, y_j)

- Representeer $V(x,y)$ in de vorm van een **matrix** :

$$V_i^j = V(x_i, y_j) \quad (i = 0, 1, 2, ..., n)(j = 0, 1, 2, ..., m)$$

- Benader de 2e afgeleiden :

$$\frac{\partial^2 V_i^j}{\partial x^2} = \frac{V_{i-1}^j - 2V_i^j + V_{i+1}^j}{h^2}$$

$$\frac{\partial^2 V_i^j}{\partial y^2} = \frac{V_i^{j-1} - 2V_i^j + V_i^{j+1}}{h^2}$$

- De Laplace vergelijking resulteert dan in een stelsel van $(n-1)(m-1)$ lineaire vergelijkingen :

$$V_{i-1}^j - 2V_i^j + V_{i+1}^j + V_i^{j-1} - 2V_i^j + V_i^{j+1} = 0$$

Hieruit volgt :

$$\boxed{V_i^j = \frac{1}{4}\left(V_{i-1}^j + V_{i+1}^j + V_i^{j-1} + V_i^{j+1}\right)} \tag{7.12}$$

M.a.w. de potentiaal in $elk\ punt$ moet gelijk zijn aan het gemiddelde van de potentialen in zijn $naaste\ buur\text{-}punten$.

- Dit stelsel vergelijkingen kan (weer) worden opgelost met behulp de methode van **Gauss-Seidel iteratie**. De methode gaat nu als volgt :
  - Zet de waarden van V_0^j V_n^j V_i^0 V_i^m vast op de randvoorwaarden. Deze waarden mogen verder **niet** veranderd worden.
  - Zet de waarde van V_i^j in de overige punten op een willekeurige startwaarde, bijv: $V_i^j = 0$.
  - Loop vervolgens de roosterpunten een voor een af,
    $(j = 1; i = 1, 2, ..., n-1)$
    $(j = 2; i = 1, 2, ..., n-1)$
    $(.....)$
    $(j = m-1; i = 1, 2, ..., n-1)$

    en vervang de 'oude waarde' van V_i^j door een 'nieuwe waarde'

$$\boxed{vervang\ V_i^j\ door\quad \frac{1}{4}\left(V_{i-1}^j + V_{i+1}^j + V_i^{j-1} + V_i^{j+1}\right)} \tag{7.13}$$

    eventueel kan ook de relaxatie parameter worden gebruikt :

$$\boxed{vervang\ V_i^j\ door\quad (1-\omega)V_i^j + \frac{\omega}{4}\left(V_{i-1}^j + V_{i+1}^j + V_i^{j-1} + V_i^{j+1}\right)} \tag{7.14}$$

  - De nieuwe set waarden voor V_i^j zal **beter** aan de voorwaarden voldoen, maar mogelijkerwijze nog **niet goed genoeg**. Vandaar dat we de vervangingsprocedure een aantal malen herhalen. Als alles goed gaat zal het resultaat convergeren naar de juiste oplossing.
  - we hebben hier weer te maken met een $iteratieve$ procedure, en hebben derhalve behoefte aan een $convergentie\text{-}monitor$: een getal dat wordt gebruikt om de iteratie te beeindigen. een geschikte grootheid hiervoor is (naar analogie van het 1-dimensionale geval) :

$$\boxed{E = \sum_{i=1}^{n}\sum_{j=1}^{m}\left[\left(V_i^j - V_{i-1}^j\right)^2 + \left(V_i^j - V_i^{j-1}\right)^2\right]} \tag{7.15}$$

7.5 de opgave : electrostatische problemen

7.5.1 de fysische achtergrond

Deze opgave betreft een eenvoudig probleem uit de Electrostatica. Het probleem is zowel analytisch als numeriek oplosbaar. Het ziet er als volgt uit:

- Beschouw een rechthoek in het xy-vlak,
  begrensd door de punten (0,0) (a,0) (0,b) en (a,b) .
- De randcondities voor de potentialen zijn:
  $V = 0$ voor $x = 0$
  $V = 0$ voor $x = a$
  $V = 0$ voor $y = b$
  $V = V_0$ voor $y = 0$, waarin $V_0 > 0$
- Bij deze randcondities ligt de waarde van $V(x, y)$ voor elk punt in de rechthoek vast.Deze wordt bepaald door de Laplace-vergelijking.
- Bij de praktische uitwerking kiezen we :

  $$a = b = 1. \qquad V_0 = 10.$$

  we verdelen het x- en het y- interval in 100 subintervallen.

7.5.2 de analytische oplossing: formules

Volgens bijv. Morse en Feshbach: Methods of Theoretical Physics, kan dit probleem langs analytische weg worden opgelost. Het resultaat is:

$$\boxed{V(x,y) = \frac{2V_0}{\pi} \sum_{n=1}^{\infty} F_1(n) F_2(n,x,a) \frac{F_3(n,y,a,b)}{F_4(n,b,a)}} \tag{7.16}$$

waarin:

$$\boxed{F_1(n) = \frac{1}{n}[1 - (-1)^n]}$$

$$\boxed{F_2(n,x,a) = \sin[\frac{\pi n x}{a}]}$$

$$\boxed{F_3(n,y,a,b) = \sinh[(\frac{\pi n}{a})(b-y)]}$$

$$\boxed{F_4(n,b,a) = \sinh[\frac{\pi n b}{a}]} \tag{7.17}$$

met

$$\sinh(u) = \frac{\exp(u) - \exp(-u)}{2}$$

7.5.3 de analytische oplossing: kant en klaar

Open en executeer $vana.py$. Bekijk hoe en wat er wordt geplot in de code en neem dat over als je aan de numerieke oplossing gaat werken, zodat je het grafische resultaat kunt gebruiken om de numerieke oplossing (zie verder) te controleren.

7.5.4 de analytische oplossing: de aanpak voor liefhebbers (leesstof, niet verplicht)

Ga na dat de analytische oplossing voldoet aan de randcondities voor
$x = 0$, $x = a$ en $y = b$
Bereken de analytische oplossing in alle roosterpunten. Neem daarbij het volgende in acht:

- De 'som' in de analytische oplossing gaat over een oneindig aantal termen. Dit kan uiteraard op de computer niet. Vervang daarom $\sum_{n=1}^{\infty}$ door $\sum_{n=1}^{nmax}$ met $nmax = 101$
- In het algemeen zal de bijdrage van een term aan de totale som kleiner worden, naarmate n groter wordt. Breek de sommatie af zodra de absolute waarde van de relatieve bijdrage kleiner wordt dan 10^{-20}
- Aan de analytische oplossing is eenvoudig te zien dat voor sommige waarden van n de bijdrage aan de potentiaal **altijd** gelijk aan 0 is. Reken deze termen **niet** uit!
- De waarden van F_3 en F_4 kunnen zeer groot worden. Behandel daarom F_1, F_2, F_3, F_4 met **dubbele precisie**.
- Om het resultaat te controleren heeft U minstens twee hulpmiddelen:
  - Uit de randcondities volgt een 'voorspelling' over het verloop van $V(x, y)$ als functie van y, bij constante x
  - Uit de symmetrie van de randcondities volgt een 'voorspelling' over het verloop van $V(x, y)$ als functie van x, bij constante y

7.5.5 De opdracht: de numerieke oplossing

De numerieke oplossing kan worden verkregen door middel van Gauss-Seidel iteratie, eventueel met gebruik van relaxatie.

De opdracht luidt :

- bereken voor het hierboven gedefinieerde probleem, de numerieke oplossing (en controleer je werk met het gegeven programma met de analytische oplossing).
- geef de gevonden resultaten grafisch weer.

Het is interessant om de volgende situaties numeriek door te rekenen (en grafisch weer te geven). Bedenk zelf de (rand)voorwaarden en experimenteer eventueel maar een beetje. Interpreteer het resultaat (dus vergelijk het met de theorie).

- een plaatcondensator.
- een of twee puntladingen.
- een zelfbedachte configuratie.

7.5.6 in te leveren

Lever in :

- *uw programma-code (electrostat.py)*
- *grafische weergave van de analytische en de numerieke oplossing van het eerste probleem.*
- *grafische weergave van de convergentie-monitor van het eerste probleem*
- *grafische resultaten van de extra opdrachten (indien gedaan).*

7.5.7 Appendix: twee-dimensionale arrays in Python

Bij deze opgave moet U werken met *matrices*. Een manier om dat in Python te implementeren is via twee-dimensionale *lists* (of eigenlijk een list van lists...). Een twee-dimensionale 5 bij 5 matrix kan als volgt worden geïnitialiseerd:

```
matrix = [[0 for x in xrange(5)] for x in xrange(5)]
```

En de elementen worden bijvoorbeeld als volgt gebruikt:

```
matrix[0][0] = 1
matrix[4][5] = 0

print matrix[0][0]
print matrix[4][5]
```

Er bestaan Python packages die geavanceerde alternatieven kunnen leveren. Het voordeel is dat er functies zijn gedefinieerd die de manipulatie van de matrices makkelijker maken. Het pakket **numpy**, bevat de *array* klasse. Zie bijvoorbeeld `http://docs.scipy.org/doc/numpy/reference/`.

8 Monte-Carlo Methoden

Een kernbergrip bij Monte-Carlo methoden is het $toevalsgetal$ (in het Engels: $random\ number$). Een toevalsgetal is een getal waarvan de waarde die het krijgt onvoorspelbaar is, zoals het werpen van een dobbelsteen. We spreken dan ook van het 'werpen' van een toevalsgetal. We kunnen wel spreken van een kansverdeling van de mogelijke waarden. Bij herhaaldelijk werpen zal deze verdeling gereproduceerd worden. Een voorbeeld van een verdeling is de $vlakke\ verdeling$ op een bepaald interval. Elke getal in dit interval heeft dan een gelijke kans om geworpen te worden.
Echte $toevalligheid$ is moeilijk te realiseren in een deterministisch systeem als een computer. De kwaliteit van een generator van toevalsgetallen wordt gemeten met de hoeveelheid worpen die men kan doen voordat de reeks zich gaat herhalen.

Monte-Carlo methoden maken gebruik van toevalsgetallen om uiteenlopende op problemen te lossen of the bestuderen. Dit wordt gedaan door de faseruimte dat het probleem beslaat te bemonsteren met gebruik van toevalsgetallen. De techniek wordt vooral gebruikt wanneer het probleem complex is; het heeft bijvoorbeeld te veel vrije parameters, ingewikkelde randvoorwaarden of er is geen analytische beschrijving voorhanden. Aan de basis van Monte-Carlo studies ligt vaak een simulatie van het probleem die het stochastisch gedrag beschrijft.

8.1 Hit or Miss

Een universeel toepasbare Monte-Carlo techniek is de zogenaamde $Hit\ or\ Miss$ methode. Deze kan bijvoorbeeld gebruikt worden voor het genereren van toevalsgetallen volgens eens arbitare verdeling of voor het integreren van een willekeurige functie.
Gegeven een functie $f(x)$ met $a \leq x \leq b$ en $f(x) \geq 0$ in dat domein, is de methode als volgt samen te vatten:

1. Werp een toevalsgetal x uit een vlakke verdeling tussen a en b
2. Bereken $f = f(x)$
3. Werp nog een toevalsgetal r, vlak, maar dit keer tussen 0 en m, waar $m \geq max(f(x))$
4. Als $r > f$, neem een nieuw toevalsgetal x
   Als $r < f$, accepteer de waarde van x

Op deze manier zal de dichtheid van x evenredig zijn aan $f(x)$

Deze methode heeft zijn beperkingen en kan inefficiënt zijn voor bijvoorbeeld hele steile functies en verdelingen. In dat geval is het beter een andere techniek te gebruiken, vooral als de functie inverteerbaar is.

8.2 Opdracht 1: Werpen van toevalsgetallen

- Schrijf een functie die toevalsgetallen genereert volgens een vlakke verdeling tussen -5 en 5. Doe dit een paar keer en bereken het gemiddelde en de standaard deviatie. Plot een histogram van de toevalsgetallen. Gebruik de functie **random** (zie beneden).

- Schrijf een functie die toevalsgetallen werpt volgens een normaal-verdeling met een gemiddelde waarde van 0 en een standaard deviatie van 1. Gebruik de $Hit\ or\ Miss$ methode. Controleer de positie van de piek een de breedte van de verdeling. Plot een histogram van de toevalsgetallen. Gebruik de functie **random** (zie beneden).

Hint: de python module **random** bevat de functie **random()** die een toevalsgetal werpt uit een vlakke verdeling op $[0, 1)$. De begin-toestand van deze (pseudo-)randomnumber generator kan gezet worden via de functie **random.seed([x])**. Zonder argument wordt de systeem tijd gebruikt (en dus is het resultaat moeilijker reproduceerbaar). Om een reproduceerbaar resultaat te maken moet een waarde van x gekozen worden. Het volgende stukje code produceert en print een toevalsgetal:

```
import random

"""
zet seed op systeemtijd
"""
random.seed()

toevalsgetal = random.random()

print toevalsgetal
```

8.3 Opdracht 2: numerieke integratie

- Bereken π door punten (x,y) te genereren in een vlak met $-1 < x < 1$ en $-1 < y < 1$. Tel het aantal keer dat een punt binnen een cirkel met straal 1 valt. Hoe verhoudt zich dat tot de ratio van de oppervlakken van de cirkel en het vierkant?

- Bereken de integraal van een normaalverdeling met gemiddelde $\mu = 0$ en breedte $\sigma = 1$ tussen de grenzen -1 en 1. Dus:

$$\int_{-1}^{1} Gauss(x|\mu, \sigma)dx \tag{8.1}$$

  Wat zou je verwachten?

- Plot de waarde van de integraal als functie van het aantal punten N (Zorg dat de punten in de grafiek onafhankelijk zijn van elkaar). Bevestig dat de relatieve fout kleiner wordt met $1/\sqrt{(N)}$.

8.4 Opdracht 3: dronkenmans wandeling

1. Schrijf een functie dat een deeltje verplaatst van een plek op een vlak naar de ander, waarbij de nieuwe positie door het toeval bepaald wordt. Maak twee versies, met

   - Een vaste stapgrootte $r = 1$, of
   - Een toevallige stapgrootte tussen 0 en 1

   Maak een grafische weergave van het pad dat een deeltje aflegt voor beide gevallen. Kies een redelijk aantal stappen.

2. Laat $n = 100$ deeltjes bewegen gedurende $m = 10$ stappen. Bereken de gemiddelde afstand tot het beginpunt $(0, 0)$. Bereken ook de standaard deviatie op deze afstand. Vergroot het aantal stappen tot $m = 1000$. Geen grafische output vereist.

3. Maak nu een simulatie voor deeltjes in een doos. Pas de functie uit 1 aan zodat de deeltjes binnen de doos blijven en aan de randen worden gereflecteerd. Kies een toevallige positie binnen de doos. Maak een grafische weergave van het pad van een deeltje. Herhaal oefening 2 voor een doos van 20 bij 20. (dus, bereken de gemiddelde afstand en standaard deviatie van 100 deeltjes na 10 en 1000 stappen).

4. Deel de doos uit 3 in twee helften (links en rechts). Begin met alle deeltjes in het linker deel. Bereken het aantal deeltjes in het rechter deel als functie van iteratie stap. Bereken de relaxatie tijd van het systeem (hier gedefinieerd als een evenwicht tussen het aantal deeltjes links en rechts.) Speel met het aantal deeltjes en aantal stappen.

5. (BONUS) Herhaal de simulatie en bereken hoe lang het duurt voordat het eerste deeltje in het rechter deel van de doos komt.

6. (BONUS) Bedenk een strategie om interacties tussen de deeltjes in rekening te nemen.

8.5 Opdracht 4: Het file-probleem

Het Nagel-Shreckenberg model is een simpele Monte-Carlo simulatie die de een aantal belangrijke eigenschappen van wegverkeer kan tonen. In het model bewegen autos op een ronde baan. De baan is opgedeeld in M zones en er zijn N wagens. De tijd verloopt in discrete eenheden. Per tijdseenheid zal een auto met snelheid v zich v zones verplaatsen. Er geldt een maximumsnelheid v_{max}. De snelheid wordt gemeten in het aantal zones dat per tijdsstap wordt afgelegd.
In elke tijdsstap gaat elke auto door de volgende fases:

1. Als de snelheid v beneden v_{max} is, versnelt de bestuurder met 1 eenheid.

2. De bestuurder bepaalt de afstand d tot de volgende auto. Als $v \geq d$ dan verlaagt de bestuurder de snelheid tot $d - 1$, teneinde niet te botsen.

3. Als $v > 0$ dan verlaagt de bestuurder met waarschijnlijkheid p de snelheid met een eenheid. Deze stap introduceert het toevallige gedrag van de bestuurders.

4. De autos nemen hun nieuwe posities in.

Deze stappen gebeuren tegelijkertijd voor alle bestuurders.
Opdrachten:

1. Implementeer het Nagel-Schreckenberg model voor een ronde baan. Vind een passende visualisatie van het resultaat.

2. Varieer N,M en v_{max}. Bereken de gemiddelde snelheid als functie van de dichtheid van autos.

3. Bedenk een passende definitie van een file en bereken de posities van eventuele files voor elke tijdsstap. Bereken de gemiddelde snelheid van files. Hoe varieert deze als functie van de variabelen?

8.6 in te leveren

Lever in:

- uw programma codes (randomnumber.py, mcintegrate.py, randomwalk.py, traffic.py)

- grafische resultaten voor elke opdracht

9 En nu zelf een probleem oplossen: Rutherford scattering

Bij deze opdracht kan er gekozen worden uit de basisopdracht **of** de uitgebreide opdracht. De uitgebreide opdracht kan maximaal een bonuspunt opleveren.
Bij beide opdrachten dient extra aandacht geschonken te worden aan de visualisatie van de resultaten en de mogelijkheid tot invoeren van keuzes voor de begincondities.

9.1 Basisopdracht

We schieten een geladen deeltje op een electrische potentiaal (bijvoorbeeld die van een ander geladen deeltje met dezelfde lading). Kies zelf redelijke en leuke begincondities (plaats en snelheid) en bereken de twee-dimensionale baan van het testdeeltje en laat dit langs grafische weg duidelijk zien. Vergelijk je resultaat met de analytische oplossing.

Bedenk eerst zelf hoe je dit probleem numeriek gaat oplossen en ga dan pas aan de slag met de code.

Literatuur: zoek het zelf maar op. Bijvoorbeeld 'Analytical mechanics van Fowles en Cassiday' blz 266.

Zorg ervoor dat je de lading en massa van je testdeeltje kunt instellen.

Wat gebeurt er allemaal als je de lading van het deeltje omkeert?

9.2 Uitgebreide opdracht

Er zijn verschillende mogelijkheden om deze opdracht uitdagender te maken.

- Het probleem uitbreiden naar 3 dimensies en 3 (of meer!) deeltjes. Bedenk interessante begincondities voor het systeem.
- Een gravitationeel systeem kan beschouwd worden, bijvoorbeeld de aarde, maan en een satelliet.
- De paden van de deeltjes kunnen middels matplotlib geanimeerd worden.

9.3 in te leveren

Lever in :

- uw programma code (scatter.py)
- grafische resultaten. Let op, hier wordt bij deze opdracht extra op gelet.

10 Partiele Differentiaal Vergelijkingen (PDE): parabolisch

Voordat je begint met lezen kun je de keuze maken om een vrijere (keuze-)opdracht te kiezen. Je kunt dus ook een van de onderstaande opdrachten kiezen, maar het is dan wel de bedoeling dat je hier zelfstandig en dus bijna zonder hulp van de assistenten uitkomt.

- **De snelheidsverdeling van Maxwell uit de kinetische gas-theorie. (goed te doen)**
  N.B. Het simpelweg implementeren van de techniek in het paper is voor een goede waardering niet voldoende. Breid deze opdracht uit, bijvoorbeeld door een dimensie toe te voegen of naar eigen inzicht.
- **Geladen deeltjes gevangen in de Van Allen gordels. (redelijk pittig)**
- **Een Quantum mechanisch golfpakketje tegen een potentiaal berg. (best moeilijk)**

Van deze onderwerpen kun je meer info vinden op de webpagina van numerieke natuurkunde (of je gaat zelf op zoek). Als je zelf suggesties voor een opdracht hebt, bespreek die dan eerst met de docent.

10.1 inleiding/probleemstelling

Typische voorbeelden van $parabolische$ partiele differentiaalvergelijkingen, zijn de $bewegings$-vergelijkingen, zoals we die in de Natuurkunde tegenkomen, en die de ruimtelijke ontwikkeling van een systeem beschrijven als funktie van de tijd. Een eenvoudig voorbeeld (niet direct met fysische betekenis) is :

- de functie $\phi(x,t)$ is gedefinieerd op het interval

$$x_{min} \leq x \leq x_{max}$$

- De waarde van de functie op het tijdstip $t = t_0$ is **gegeven**
- De ontwikkeling van $\phi(x,t)$ in de tijd wordt beschreven door

$$\boxed{\frac{\partial \phi}{\partial t} = \frac{\partial^2 \phi}{\partial x^2} + S(x)} \tag{10.1}$$

- Bovendien moet tijdens de ontwikkeling in de tijd aan de volgende **randcondities** worden voldaan

$$\phi(x_{min}, t) = \phi_{min} = constant$$

$$\phi(x_{max}, t) = \phi_{max} = constant \tag{10.2}$$

- Gevraagd wordt om de waarde van $\phi(x,t)$ op een willekeurig tijdstip te berekenen.

10.2 discretisatie van ruimte en tijd

- Ruimtelijke discretisatie
  - Verdeel interval (x_{min}, x_{max}) in N intervallen van gelijke lengte h . Coordinaten van de intervalgrenzen (roosterpunten) zijn :

$$x_j = x_{min} + jh \qquad (j = 0, 1, 2....., n)$$

  - Representeer $\phi(x)$ en $S(x)$ in de vorm van een **tabel**

$$\phi_j = \phi(x_j) \quad S_j = S(x_j) \qquad (j = 0, 1, 2, ..., n)$$

  - Benader de 2e afgeleide :

$$\frac{\partial^2 \phi_j}{\partial x^2} = \frac{\phi_{j-1} - 2\phi_j + \phi_{j+1}}{h^2}$$

- Discretisatie van de tijd
  - Besluit om $\phi(x, t)$ uit te rekenen in tijdstapjes Δt , d.w.z. op tijdstippen

$$t_k = t_0 + k(\Delta t) \qquad (k = 0, 1, 2, ...)$$

  - De waarden van $\phi(x, t)$ op de roosterpunten na de k-de tijdstap noteren we als

$$\phi_j^k \qquad (j = 0, 1, 2, ..., n) \qquad (k = 0, 1, 2, ...)$$

  - Via de begincondities zijn vastgelegd :

$$\phi_j^0 \qquad (j = 0, 1, 2, ..., n)$$

  - De randcondities houden in

$$\phi_0^k = \phi_0^0 \qquad \phi_n^k = \phi_n^0 \qquad \forall k$$

10.3 de expliciete methode

- In deze methode benaderen we de tijdsafgeleide door de voorwaartse 2-punts formule

$$\frac{\partial \phi_j^k}{\partial t} = \frac{\phi_j^{k+1} - \phi_j^k}{\Delta t}$$

- De basisvergelijking resulteert dan in een stelsel van $(n + 1)$ vergelijkingen

$$\frac{\phi_j^{k+1} - \phi_j^k}{\Delta t} = \frac{\phi_{j-1}^k - 2\phi_j^k + \phi_{j+1}^k}{h^2} + S_j \tag{10.3}$$

- De oplossing hiervan is

$$\phi_j^{k+1} = \frac{\Delta t}{h^2}\left(\phi_{j-1}^k + \phi_{j+1}^k\right) + \left(1 - 2\frac{\Delta t}{h^2}\right)\phi_j^k + S_j\Delta t \tag{10.4}$$

- Je kunt dit (wederom) interpreteren als een matrix-vergelijking:

$$\boxed{\vec{\phi}^{k+1} = (I - H\Delta t)\vec{\phi}^k + \vec{S}\Delta t} \tag{10.5}$$

waarin I de eenheidsmatrix is en waarin H een **matrix** is met elementen

$$H = \frac{1}{h^2}\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & ... & 0 \\ -1 & 2 & -1 & 0 & 0 & ... & 0 \\ 0 & -1 & 2 & -1 & 0 & ... & 0 \\ . & . & . & . & . & ... & . \\ . & . & . & . & . & ... & . \\ . & . & . & . & . & ... & . \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & ... & 0 \end{pmatrix} \tag{10.6}$$

- Wanneer we de tijdstap Δt in samenhang met de roosterafstand h zodanig kiezen dat $2\Delta t = h^2$ vereenvoudigt de oplossing tot

$$\phi_j^{k+1} = \frac{1}{2}\left(\phi_{j-1}^k + \phi_{j+1}^k\right) + S_j\Delta t \tag{10.7}$$

- Onafhankelijk van de keuze van Δt en h is de berekening **eenvoudig** uit te voeren.

- **Echter** : in de praktijk blijkt dat deze methode slechts convergeert naar een stabiele oplossing, wanneer Δt in samenhang met h zodanig gekozen wordt dat

$$\frac{\Delta t}{h^2} \leq \frac{1}{2}$$

Dit is een **ongewenste** beperking.

10.4 een impliciete methode

- In deze methode benaderen we de tijdsafgeleide door de achterwaartse 2-punts formule

$$\frac{\partial \phi_j^k}{\partial t} = \frac{\phi_j^k - \phi_j^{k-1}}{\Delta t}$$

- De basisvergelijking resulteert dan in

$$\frac{\phi_j^{k+1} - \phi_j^k}{\Delta t} = \frac{\phi_{j-1}^{k+1} - 2\phi_j^{k+1} + \phi_{j+1}^{k+1}}{h^2} + S_j \tag{10.8}$$

- In matrix-notatie :

$$\vec{\phi}^{k+1} - \vec{\phi}^{k} = (-H\Delta t)\vec{\phi}^{k+1} + \vec{S}\Delta t \tag{10.9}$$

de oplossing hiervan is

$$\boxed{\vec{\phi}^{k+1} = (I + H\Delta t)^{-1}\left[\vec{\phi}^{k} + \vec{S}\Delta t\right]} \tag{10.10}$$

waarin de matrices I en H gedefinieerd zijn als bij de expliciete methode.

- In de praktijk blijkt dat deze methode niet onderhevig is aan de beperking van de expliciete methode omtrent de keuze van Δt en h.

10.5 de Crank-Nicholson methode

- Deze methode (die ook impliciet is) kiest het **midden** tussen beide voorgaande methodes door te starten met de volgende basisvergelijking

$$\vec{\phi}^{k+1} = \vec{\phi}^{k} - \frac{1}{2}H\Delta t\left(\vec{\phi}^{k+1} + \vec{\phi}^{k}\right) + \vec{S}\Delta t \tag{10.11}$$

- De oplossing hiervan is

$$\boxed{\vec{\phi}^{k+1} = \left(I + \frac{1}{2}H\Delta t\right)^{-1}\left[\left(I - \frac{1}{2}H\Delta t\right)\vec{\phi}^{k} + \vec{S}\Delta t\right]} \tag{10.12}$$

- In de praktijk blijkt dat ook deze methode niet onderhevig is aan de beperking van de expliciete methode.

10.6 oplossingstechniek voor Crank Nicholson

Het is uiteraard mogelijk om vgl (10.12) direct op te lossen als matrix-vergelijking. Dit vereist echter de constructie en inversie van een $(n+1) \star (n+1)$ matrix.
Bovendien heeft de matrix een bijzondere structuur: **tridiagonaal**.
Vandaar dat andere oplossingsrecepten efficienter kunnen zijn (in termen van geheugenruimte en/of computertijd)
We behandelen de methode van **Gaussische eliminatie en terug-substitutie**.

- Vergelijking (10.12) kan als volgt herschreven worden

$$\vec{\phi}^{k+1} = \left(I + \frac{1}{2}H\Delta t\right)^{-1}\left[-\left(I + \frac{1}{2}H\Delta t\right)\vec{\phi}^{k} + 2\vec{\phi}^{k} + \vec{S}\Delta t\right] \tag{10.13}$$

- dit is te schrijven als :

$$\boxed{\vec{\phi}^{k+1} = \vec{\chi}^{k} - \vec{\phi}^{k}} \tag{10.14}$$

met :

$$\boxed{\left(I + \frac{1}{2}H\Delta t\right)\vec{\chi}^k = 2\vec{\phi}^k + \vec{S}\Delta t} \tag{10.15}$$

De termen in het rechterlid van deze vergelijking zijn bekende grootheden (voor $k = 0$ uit de randcondities). De matrix in het linkerlid is ook bekend.
Resteert de berekening van $\vec{\chi}^k$.

- vergelijking (10.15) is door de tridiagonaliteit van H te schrijven als een stelsel van $(n-1)$ lineaire vergelijkingen:

$$\boxed{A_j^+\chi_{j+1} + A_j^0\chi_j + A_j^-\chi_{j-1} = b_j} \tag{10.16}$$

met

$$A_j^+ = A_j^- = -\frac{\Delta t}{2h^2} \qquad A_j^0 = 1 + \frac{\Delta t}{h^2} \qquad b_j = 2\phi_j^k + S_j\Delta t \tag{10.17}$$

- Gaussische eliminatie en terugsubstitutie komt er op neer dat we een oplossing zoeken waarbij χ voldoet aan een **voorwaartse recursie-relatie**

$$\boxed{\chi_{j+1} = \alpha_j\chi_j + \beta_j} \tag{10.18}$$

- Substitutie van vgl (10.18) in vgl (10.16) en 'herschikking' levert

$$\chi_j = \gamma_j A_j^-\chi_{j-1} + \gamma_j(A_j^+\beta_j - b_j) \tag{10.19}$$

met

$$\boxed{\gamma_j = -\left(A_j^0 + A_j^+\alpha_j\right)^{-1}} \tag{10.20}$$

Hiermee hebben we voor de coefficienten α en β een **achterwaartse recursie**-relatie verkregen

$$\boxed{\alpha_{j-1} = \gamma_j A_j^- \qquad \beta_{j-1} = \gamma_j\left(A_j^+\beta_j - b_j\right)} \tag{10.21}$$

- Omdat de randvoorwaarden eisen dat ϕ_0 en ϕ_n niet mogen veranderen moet gelden dat

$$\chi_n = \alpha_{n-1}\chi_{n-1} + \beta_{n-1} = 2\phi_n \tag{10.22}$$

hieraan is alleen voldaan als

$$\alpha_{n-1} = 0 \qquad \beta_{n-1} = 2\phi_n \tag{10.23}$$

- het 'recept' om $\vec{\phi}^{k+1}$ te berekenen uit $\vec{\phi}^k$ gaat als volgt
  - Start met

$$\alpha_{n-1} = 0 \qquad \beta_{n-1} = 2\phi_n \qquad \gamma_{n-1} = -\left(A_{n-1}^0 + A_{n-1}^+\alpha_{n-1}\right)^{-1}$$

– Bereken via de achterwaartse recursie-relaties (10.21) de overige coefficienten α_j en β_j gebruikmakend van de hulpgrootheid γ_j uit vgl (10.20)

– Bereken met de voorwaartse recursie-formule (10.18) de grootheden

$$\chi_j \qquad (j = 1, 2, ..., n-1)$$

Dit zijn de gezochte waarden van $\vec{\chi}_j^k$

– Bereken de gezochte waarden van $\vec{\psi}^{k+1}$ met behulp van vgl (10.14)

10.7 eenvoudige opgave

Om vertrouwd te raken met expliciete versus impliciete methoden, en met de techniek van Gaussische eliminatie en terug-substitutie beginnen we met het volgende probleem

- Vergelijking

$$\boxed{\frac{\partial \phi}{\partial t} = \frac{\partial^2 \phi}{\partial x^2} \qquad op \quad interval \qquad 0 \leq x \leq 1}$$

- Randcondities

$$\boxed{\phi(0,t) = \phi(1,t) = 0}$$

- Vorm van $\phi(x,0)$

$$\boxed{\sum_{j=-1}^{1} (-1)^j \exp\left[-20\left(x - j - \frac{1}{2}\right)^2\right]}$$

- Analytische oplossing voor $\phi(x,t)$

$$\frac{1}{\sqrt{\tau}} \sum_{j=-1}^{1} (-1)^j \exp\left[-\frac{20}{\tau}\left(x - j - \frac{1}{2}\right)^2\right] \qquad met \qquad \tau = 1 + 80t$$

DE **OPDRACHT** luidt:

- Maak een programma dat $\phi(x,t)$ berekent met de **expliciete** methode.
- Maak een programma dat $\phi(x,t)$ berekent met de **Crank-Nicholson** methode.
- Verifieer de kwaliteit van de oplossing van beide methoden.

Praktische zaken :

- Verdeel het x-interval in 100 sub-intervallen.
- Laat de tijd lopen van $t = 0$ tot $t = 0.0048$.
- Gebruik voor de expliciete methode de volgende twee tijdstappen : $\Delta t = 0.00004$ en $\Delta t = 0.00006$
- Gebruik voor de impliciete methode de volgende drie tijdstappen : $\Delta t = 0.00004$, $\Delta t = 0.00006$ en $\Delta t = 0.0006$
- Gebruik grafische weergave ter inspectie van $\phi(x,t)$

10.7.1 in te leveren

Lever in :

- *uw programma-code*
- *grafische resultaten*

10.8 de opgave : een radio-aktief afval probleem

In deze opgave houden we ons bezig met een probleem uit de 'wereld' van de opslag van radio-aktief afval.Daarbij moet niet alleen de 'omgeving' worden beschermd tegen de vrijkomende straling, maar er moet ook rekening worden gehouden met het feit dat de vrijkomende straling een $verhoging\ van\ de\ temperatuur$ veroorzaakt: de beschermende materialen moeten hiertegen bestand zijn.

10.8.1 de fysische achtergrond

We beschouwen een (oneindig lange) cylindervormige staaf van radio-aktief materiaal. Voor de temperatuur $T(r,t)$ op afstand r op tijdstip t geldt de volgende 'diffusie' vergelijking :

$$\boxed{\frac{1}{\kappa}\frac{\partial T(r,t)}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}r\frac{\partial T(r,t)}{\partial r} + S(r,t)} \tag{10.24}$$

waarin κ de diffusie constante is.
Een geschikte vorm voor de 'bronterm' $S(r,t)$ ten gevolge van het radio aktieve verval van de cylindervormige staaf is :

$$\boxed{S(r,t) = \frac{T_0}{a^2}\exp\left(-\frac{t}{\tau_0}\right) \qquad r \leq a} \tag{10.25}$$

hierin zijn :

a	de straal van de staaf
T_0	de 'temperatuur' van de staaf op tijdstip $t = 0$
τ_0	de 'vervalsconstante' van het radioaktieve materiaal.

10.8.2 de numerieke oplossing

we kunnen vgl (10.24) als volgt schrijven

$$\boxed{\frac{\partial T(r,t)}{\partial t} = \frac{\kappa}{r}\left[\frac{\partial T(r,t)}{\partial r} + r\frac{\partial^2 T(r,t)}{\partial r^2}\right] + \kappa S(r,t)} \tag{10.26}$$

er zijn twee verschillen met de eerdere vgl (10.1) :

- er komt nu ook een 1ste afgeleide naar de plaats in voor.
- de 'bronterm' $S(r,t)$ is nu tijdsafhankelijk

om de vergelijking geschikt te maken voor de numerieke aanpak, discretiseren we plaats en tijd :

- Verdeel interval (r_{min}, r_{max}) in N intervallen van gelijke lengte h .
  Coordinaten van de intervalgrenzen (roosterpunten) zijn :

$$r_j = r_{min} + jh \qquad (j = 0, 1, 2....., n)$$

- Besluit om $T(r,t)$ uit te rekenen in tijdstapjes Δt , d.w.z. op tijdstippen

$$t_k = t_0 + k(\Delta t) \qquad (k = 0, 1, 2, ...)$$

- Representeer $T(r,t)$ en $S(r,t)$ in de vorm van **tabellen**

$$T_j^k = T(r_j, t_k) \quad S_j^k = S(r_j, t_k) \qquad (j = 0, 1, 2, ..., n) \quad (k = 0, 1, 2, ...)$$

- Benader de 1e ruimtelijke afgeleide :

$$\frac{\partial T_j^k}{\partial r} = \frac{T_{j+1}^k - T_{j-1}^k}{2h}$$

- Benader de 2e ruimtelijke afgeleide :

$$\frac{\partial^2 T_j^k}{\partial r^2} = \frac{T_{j-1}^k - 2T_j^k + T_{j+1}^k}{h^2}$$

- Via de begincondities zijn vastgelegd :

$$T_j^0 \qquad (j = 0, 1, 2, ..., n)$$

- De randcondities houden in

$$T_0^k = T_0^0 \qquad T_n^k = T_n^0 \qquad \forall k$$

voor de discretisatie van de tijd (in samenhang met de plaats) kiezen we voor Crank-Nicholson methode. we lossen het stelsel vergelijkingen dat aldus ontstaat, op met Gaussische eliminatie en terugsubstitutie. het geheel is beschreven in sectie 7.6. het enige verschil treedt op bij vgl (10.17) ; de nieuwe definities van de daar genoemde grootheden zijn :

$$\boxed{A_j^+ = -\frac{\kappa \Delta t}{2h^2}\left(1 + \frac{h}{2r_j}\right) \qquad A_j^- = -\frac{\kappa \Delta t}{2h^2}\left(1 - \frac{h}{2r_j}\right) \qquad A_j^0 = 1 + \frac{\kappa \Delta t}{h^2}}$$

$$\boxed{b_j = 2T_j^k + \frac{\kappa \Delta t}{2}\left(S_j^k + S_j^{k+1}\right)} \tag{10.27}$$

10.8.3 de opdracht

de opdracht luidt :

- bereken $T(r,t)$ na 5, 50, 100 en 200 jaar.
- geef resultaat grafisch weer.

Uiteraard moet je wat redelijke begincondities opstellen. Je zou de onderstaande 'waarden' kunnnen gebruiken, maar je kunt ook zelf wat experimenteren:

- zet de temperatuur op $(t = 0)$ gelijk aan 0. **(overal, ook binnen de staaf)**.
- zet de 'extra temperatuur' van de staaf op $(t = 0)$: $T_0 = 10K$
- zet de straal van de staaf $(a = 25cm)$.
- zet de vervalsconstante $(\tau_0 = 1.)$ (in eenheden van 100 jaar)
- zet de diffusieconstante $(\kappa = 3153.6)$ (in eenheden $cm^2 per 100 jaar$).
- zet $(r_{min} = 0cm)$ en $(r_{max} = 100cm)$ en houdt de temperatuur daar **vast** op de waarde van $(t = 0)$
- verdeel het r-interval in 100 stappen.
- zet de tijdstap $(\Delta t = 0.001)$; in ons eenhedenstelsel betekent dit dat elke tijdstap correspondeert met 0.1 jaar.

10.8.4 in te leveren

Lever in :

- *uw programma-code*
- *grafische resultaten*