Onderzoek: Mogelijkheden van migratie

Fontys | TILBURG

Kayle Boersen

2024

Contents

[Matlab 3](#_Toc170063206)

[Wat is Matlab? 3](#_Toc170063207)

[Enkele voordelen van MATLAB: 3](#_Toc170063208)

[Nadelen van Matlab: 3](#_Toc170063209)

[Migratie mogelijkheden 4](#_Toc170063210)

[Voordelen: 4](#_Toc170063211)

[Nadelen: 4](#_Toc170063212)

[Libraries: 4](#_Toc170063213)

[Julia: 6](#_Toc170063214)

[Voordelen: 6](#_Toc170063215)

[Nadelen: 6](#_Toc170063216)

[Libraries: 6](#_Toc170063217)

[Conclusie 7](#_Toc170063218)

[Testen 8](#_Toc170063219)

[Matlab 8](#_Toc170063220)

[Python 9](#_Toc170063221)

[Bokeh 9](#_Toc170063222)

[Plotly 9](#_Toc170063223)

[Dash 9](#_Toc170063224)

[Conclusie 9](#_Toc170063225)

[Matrix 10](#_Toc170063226)

[Wat is een Matrix? 10](#_Toc170063227)

[Waarom worden matrixen gebruikt? 10](#_Toc170063228)

[Hoe kan een matrix gebruikt worden in Python? 10](#_Toc170063229)

[Voorbeelden 10](#_Toc170063230)

[Matrices definiëren 10](#_Toc170063231)

[Optellen 10](#_Toc170063232)

[Aftrekken 11](#_Toc170063233)

[Vermenigvuldigen 11](#_Toc170063234)

[Delen 11](#_Toc170063235)

[Factor 11](#_Toc170063236)

[Doorlopende berekening 12](#_Toc170063237)

[Grafieken 13](#_Toc170063238)

[Wat is een Grafiek? 13](#_Toc170063239)

[Hoe kan een Grafiek gebruikt worden in Python? 13](#_Toc170063240)

[Hoe hebben wij het gebruikt 13](#_Toc170063241)

[Imports 13](#_Toc170063242)

[App starten 14](#_Toc170063243)

# Matlab

## Wat is Matlab?

MATLAB (afkorting van **MAT**rix **LAB**oratory) is een softwareomgeving die gespecialiseerd is in wiskundige berekeningen en visualisatie. Het is ontwikkeld door MathWorks en wordt wereldwijd gebruikt door miljoenen engineers, wetenschappers en studenten.

## Enkele voordelen van MATLAB:

**Gebruiksvriendelijk:** De interface van MATLAB is intuïtief en gebruiksvriendelijk, waardoor het ook toegankelijk is voor beginners.

**Uitgebreide bibliotheek:** MATLAB bevat een uitgebreide bibliotheek met kant-en-klare functies voor diverse toepassingen.

**Flexibel**: MATLAB is een flexibele tool die je kunt aanpassen aan jouw specifieke behoeften.

**Compatibel met andere software:** MATLAB kan integreren met andere software, zoals C/C++, Python en Simulink.

## Nadelen van Matlab:

**Kosten:** MATLAB is een dure softwarepakket wat kan oplopen tot 900 euro per jaar. Voor sommige gebruikers is dit onbetaalbaar.

**Langzamer:** MATLAB is een geïnterpreteerde taal. Hierdoor kan het programma langzamer uitvoeren dan gecompileerde talen.

# Migratie mogelijkheden

Python:

Voordelen:

* **Groot ecosysteem:** Bibliotheken zoals NumPy, SciPy en Matplotlib bieden vergelijkbare functionaliteit als MATLAB voor numeriek rekenen, lineaire algebra en data visualisatie.
* **Open-source en gratis:** Geen licentiekosten.
* **Over het algemeen sneller:** Python kan sneller zijn dan MATLAB, vooral voor taken met grote datasets.

### Nadelen:

* **Kan minder snel zijn voor specifieke taken:** Hoewel Python over het algemeen sneller is, kan MATLAB in bepaalde gevallen (bijvoorbeeld sterk geoptimaliseerde code) nog steeds sneller zijn.
* **Leereffect:** Als u en uw team niet bekend zijn met Python, kan het even duren om de taal te leren.

### Libraries:

#### Dash:

**Voordelen:**

* Gebruiksvriendelijk & snel te leren
* Plotly-integratie voor interactieve grafieken

**Nadelen:**

* Beperkte flexibiliteit
* Afhankelijk van Plotly

*Source: <https://plotly.com/dash/>*

#### Plotly:

**Voordelen**:

* Interactieve grafieken
* Visueel aantrekkelijk
* JavaScript-bibliotheek voor webapps & notebooks

**Nadelen:**

* Moeilijkheidsgraad voor beginners
* Grote bestandsgrootte

*Source:* [*https://plotly.com/*](https://plotly.com/)

#### Matplotlib:

**Voordelen:**

* Breed gebruik & stabiliteit
* Flexibiliteit
* Werkt met andere bibliotheken

**Nadelen:**

* Steile leercurve
* Snelheidsproblemen met grote datasets

*Source:* [*https://matplotlib.org/*](https://matplotlib.org/)

#### Bokeh:

**Voordelen:**

* Real-time data visualisatie
* Interactieve grafieken
* Server-side rendering voor efficiënte verwerking

**Nadelen:**

* Moeilijkheidsgraad
* Beperkte grafiektypen

*Source:* [*https://bokeh.org/*](https://bokeh.org/)

## Julia:

### Voordelen:

* **Snelheid:** Julia wordt vaak als sneller beschouwd dan zowel Python als MATLAB voor vergelijkbare taken.
* **Gratis en open-source:** Geen licentiekosten.

### Nadelen:

* **Jonger en kleinere community:** Julia is een nieuwere taal dan Python, wat betekent dat de community kleiner is en er minder hulpbronnen beschikbaar zijn.
* **Steilere leercurve:** Julia kan een lastigere taal zijn om te leren dan Python, vooral voor beginners.

### Libraries:

#### Plotly.jl:

**Voordelen:**

* Creëert verbluffende interactieve grafieken.
* Ondersteunt een uitgebreid assortiment grafiektypen.

**Nadelen:**

* Vereist Plotly.js voor het renderen van grafieken.
* De leercurve kan steiler zijn in vergelijking met andere bibliotheken.

*Source: <https://github.com/plotly/Plotly.jl>*

#### VegaLite.jl:

**Voordelen:**

* Maakt gebruik van een declaratieve syntaxis die eenvoudig te leren en te gebruiken is.
* Biedt flexibiliteit in het definiëren van de structuur en stijl van je grafieken.
* Kan grafieken genereren voor verschillende platforms.

**Nadelen:**

* Biedt mogelijk een kleiner scala aan grafiektypen in vergelijking met andere bibliotheken.
* De focus ligt op statische grafieken, met minder interactieve mogelijkheden dan andere opties.

*Source: <https://github.com/queryverse/VegaLite.jl>*

#### Gadfly.jl:

**Voordelen:**

* Ontworpen met een gebruiksvriendelijke interface die eenvoudig te leren is.
* Produceert visueel aantrekkelijke grafieken met publicatiewaardige kwaliteit.
* Maakt integratie met LaTeX-wiskundige notaties mogelijk.

**Nadelen:**

* Richt zich op statische grafieken, met beperkte interactieve mogelijkheden.
* Biedt mogelijk een kleiner scala aan grafiektypen in vergelijking met andere bibliotheken.

*Source: <https://github.com/GiovineItalia/Gadfly.jl>*

## Conclusie

Nadat ik Python en Julia heb vergeleken voor data-analyse en visualisatie, ligt mijn voorkeur bij Python. Hier zijn de belangrijkste redenen voor mijn keuze:

Python biedt een grote selectie aan krachtige bibliotheken zoals NumPy, SciPy en Plotly, die essentieel zijn voor numerieke berekeningen, lineaire algebra en data visualisatie. Deze tools zijn vergelijkbaar met wat MATLAB biedt, maar zonder de kosten van een licentie. Python presteert over het algemeen goed bij taken met grote datasets, wat cruciaal is voor mijn studieprojecten en toekomstige werkzaamheden. Hoewel er een leercurve is, vooral bij complexere bibliotheken, is deze over het algemeen goed te overbruggen.

Julia heeft indruk op me gemaakt vanwege zijn snelheid, die vaak sneller wordt genoemd dan Python en MATLAB voor soortgelijke taken. Als open-source taal zonder licentiekosten biedt Julia zeker aantrekkelijke voordelen. Echter, de jongere gemeenschap en steilere leercurve kunnen beperkingen vormen voor mijn huidige studie- en projectbehoeften.

Daarnaast speelt populariteit een belangrijke rol in mijn keuze. Python is aanzienlijk populairder dan Julia volgens de [TIOBE Index - TIOBE](https://www.tiobe.com/tiobe-index/), waar Python op de eerste plaats staat en MATLAB op de veertiende plaats. Dit betekent dat er meer bronnen, tutorials en gemeenschapsondersteuning beschikbaar zijn voor Python. Dit maakt het eenvoudiger om hulp te vinden en problemen op te lossen, wat van onschatbare waarde is bij het leren en gebruiken van een programmeertaal. Bovendien, als dit project wordt overgedragen en er nieuwe programmeurs worden gezocht om de implementatie voort te zetten, zal het vanwege de populariteit van Python gemakkelijker zijn om gekwalificeerde ontwikkelaars te vinden.

Hoewel Julia sneller is dan Python, zijn bibliotheken die we in dit project gebruiken, zoals NumPy, geschreven in C. Dit zorgt ervoor dat de performance voor complexe berekeningen vergelijkbaar is met die van Julia, waardoor het snelheidsvoordeel van Julia minder significant is in de context van dit project.

Afbeelding met tekst, schermopname, nummer, software

Automatisch gegenereerde beschrijvingIk heb daarom besloten om verder te gaan met Python vanwege zijn brede toepasbaarheid, ondersteuning vanuit de gemeenschap, vergelijkbare performance dankzij geoptimaliseerde bibliotheken, en de mogelijkheid om snel te leren en toe te passen in mijn studie en toekomstige carrière in de ICT.

# Testen

## Matlab

Bij het testen van het bestaande MATLAB-project kwamen er meteen enkele problemen aan het licht. Om de MATLAB IDE te downloaden en gebruiken, moesten we ons eerst registreren. Als studenten konden we gelukkig gebruikmaken van een account via de school, waardoor er voor ons geen extra kosten aan verbonden waren. Voor anderen kan dit echter een drempel vormen.

Na de installatie van de IDE merkte ik dat het opstarten extreem lang duurde. De opstarttijd varieerde van 55 seconden tot wel 4 minuten. Toen de IDE eindelijk was opgestart, probeerde ik het programma zelf te starten. Dit duurde ook erg lang; in onze tests kostte het tussen de 35 en 40 seconden op een goede computer.

Toen het project volledig was opgestart, verschenen er 14 verschillende vensters, elk met andere data. Dit maakte het onmogelijk om een aanpassing te doen zonder de hele berekening opnieuw uit te voeren. In een trainingsscenario zou dit niet snel en effectief genoeg zijn om artsen in staat te stellen echte simulaties te maken. De gewenste performance is dat de simulatie snel kan worden gestart en dat er live updates aan de gegevens kunnen worden uitgevoerd, zonder 30 seconden te hoeven wachten tot de berekening klaar is.Afbeelding met schermopname, software, Multimediasoftware

Automatisch gegenereerde beschrijving

## Python

Om de mogelijkheden te testen van Python heb ik de verschillende libraries die eerder in dit document beschreven waren gepakt en deze tegen elkaar vergeleken in de praktijk.

### Bokeh

Bokeh bleek een krachtige tool voor interactieve visualisaties. De mogelijkheid om real-time data te visualiseren met server-side rendering sprak me aan, vooral omdat het geschikt is voor dynamische toepassingen. Ik merkte echter op dat de programmeerstijl soms wat uitgebreider lijkt in vergelijking met andere bibliotheken, wat de leercurve iets kan verhogen.

## Plotly

Plotly maakte indruk met zijn gebruiksvriendelijke API en de mogelijkheid om veel interactieve grafieken te creëren. De integratie met JavaScript voor rendering maakt het zeer geschikt voor webapplicaties. Ik vond het prettig om snel en effectief complexe grafieken te kunnen maken.

### Dash

Dash, gebouwd bovenop Plotly, bood een uitstekende oplossing voor het ontwikkelen van interactieve visualisaties. De integratie met Plotly stelde me in staat om snel dynamische en aanpasbare visualisaties te realiseren met een minimale hoeveelheid code. De flexibiliteit en uitbreidingsmogelijkheden van Dash maken het een ideale keuze voor het project.

### Conclusie

Na uitvoerig testen heb ik besloten om voor Dash te kiezen. Dash biedt de krachtige grafiekfunctionaliteiten van Plotly en combineert deze met uitgebreide mogelijkheden voor uitbreiding en aanpassing. Deze combinatie maakt Dash geschikt voor mijn project, waarbij ik complexe en interactieve visualisaties moet ontwikkelen.

# Matrix

Wat is een Matrix?  
Een matrix is een rechthoekig schema van getallen, ook wel elementen genoemd. De getallen zijn gerangschikt in rijen en kolommen. Het aantal rijen wordt aangeduid met m en het aantal kolommen met n. We noemen dit dan een m x n matrix.

Een matrix kan je noteren met haakjes en komma's, waarbij de getallen per rij gescheiden worden door komma's en de rijen door haakjes en puntkomma's.

Een voorbeeld van een 2 x 3 matrix:

A = [1, 2, 3;

4, 5, 6]

Hierbij is A de naam van de matrix, 1, 2, 3, 4, 5 en 6 zijn de elementen, en is m = 2 (aantal rijen) en n = 3 (aantal kolommen).

## Waarom worden matrixen gebruikt?

Matrices worden veel gebruikt in de wetenschap, techniek, wiskunde en informatica omdat ze gemakkelijk kunnen worden gebruikt om lineaire vergelijkingen te representeren of om lineaire transformaties uit te voeren. Bijvoorbeeld, in computergraphics worden matrices gebruikt om rotatie, schaalverandering, vertaling, en andere transformaties te representeren.

## Hoe kan een matrix gebruikt worden in Python?

Python biedt een module genaamd NumPy om efficiënt met matrices te werken. NumPy biedt verschillende functies om matrices te maken, te manipuleren en te analyseren.

## Voorbeelden

### Matrices definiëren

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

### Optellen

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

### Aftrekken

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

### Vermenigvuldigen

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

### Delen

A black background with colorful text

Description automatically generated

### Factor

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

Dit zijn slechts enkele voorbeelden van de vele mogelijkheden die NumPy biedt voor het werken met matrices in Python. Naast NumPy zijn er nog andere bibliotheken beschikbaar voor matrixbewerkingen in Python, zoals SciPy en TensorFlow. SciPy word gebruikt voor geavanceerde statistische analyse, signaalverwerking of optimalisatieproblemen *(SciPy, n.d.)*, terwijl TensorFlow meer gebruikt word voor het bouwen en trainen van complexe machine learning-modellen, met name deep learning-modellen *(Tensorflow, n.d.)*.

## Doorlopende berekening

Als we gebruik gaan maken van Matrix calculaties om continue berekeningen te maken moet het mogelijk zijn om de calculatie dus op langertermijn uit te voeren. Daarvoor kunnen we gebruik maken van een “While True” script. In deze “While true” zal alles steeds opnieuw gerund worden.

Afbeelding met tekst, schermopname, Lettertype, software

Automatisch gegenereerde beschrijving

Om ervoor te zorgen dat we niet steeds dezelfde data genereren heb ik erin gezet dat Matrix A en Matrix B steeds verschillende waardes hebben.

Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving

Om de snelheid van de calculaties aan te passen kunnen we het getal bij “sleep\_timer” aanpassen. Op deze manier kunnen de dus vragen aan het script om sneller of juist langzamer de calculaties te maken. Vanuit Beatrijs was de vraag om minimaal 1 datapunt te genereren per seconde. Hier zien we dat het mogelijk is vanuit de Matrix kant. Later zullen we onderzoeken of het ook mogelijk is vanuit de grafiek kant.

# Grafieken

## Wat is een Grafiek?

Een grafiek is een visuele weergave van data die helpt om patronen, trends en relaties te begrijpen. Door data in grafieken te plotten, kunnen we complexe gegevens op een eenvoudige en begrijpelijke manier presenteren. Er zijn verschillende soorten grafieken, zoals lijngrafieken, staafdiagrammen, cirkeldiagrammen en scatterplots. Elk type grafiek heeft zijn eigen unieke voordelen en toepassingen. In ons project zullen we gebruik maken van een lijngrafiek, aangezien deze het beste de verandering van data over tijd laat zien.

## Hoe kan een Grafiek gebruikt worden in Python?

In Python zijn er verschillende bibliotheken beschikbaar om grafieken te maken en te manipuleren. De meest gebruikte bibliotheek is Matplotlib, die krachtige functies biedt voor het maken van allerlei soorten grafieken. Een andere populaire optie is Seaborn, die is gebouwd op Matplotlib en extra functionaliteit biedt voor het maken van statistische grafieken.

Voor dit project zullen we gebruik maken van Dash, aangezien deze speciaal ontworpen is voor het bouwen van interactieve webapplicaties met grafieken. Dash maakt gebruik van Plotly om dynamische en responsieve grafieken te creëren die gemakkelijk te integreren zijn in webapplicaties.

## Hoe hebben wij het gebruikt

### Imports

We beginnen met het importeren van de benodigde bibliotheken voor het ontwikkelen van onze Dash-applicatie.

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

Als eerste importeren we Dash zelf met “import dash”. Dash is een bibliotheek in Python die wordt gebruikt voor het bouwen van interactieve webapplicaties. Het stelt ons in staat om data-analyses en visualisaties te integreren in een webgebaseerde omgeving met behulp van Python.

Daarna importeren we “dash\_core\_components” en “dash\_html\_components” als “dcc” en “html”. Deze modules zijn essentieel voor het definiëren van de gebruikersinterface binnen een Dash-applicatie. dcc biedt componenten zoals grafieken, dropdowns, sliders en meer, terwijl html wordt gebruikt voor het opzetten van de structuur en lay-out van de webpagina's met HTML-elementen zoals divs, headers en paragrafen.

Vervolgens importeren we “plotly.graph\_objs” als “go”. Plotly is een bibliotheek voor interactieve datavisualisatie en biedt grafische objecten (graph objects) waarmee we complexe grafieken en visualisaties kunnen maken binnen onze Dash-applicatie. Door go te importeren, hebben we toegang tot functies en objecten waarmee we grafieken kunnen aanpassen en interactieve elementen kunnen toevoegen aan onze visualisaties.

Tot slot, om onze Dash-applicatie te initialiseren, maken we een object genaamd “app” aan met “dash.Dash(\_\_name\_\_)”. Hierbij is “\_\_name\_\_” een speciale variabele die de naam van het huidige Python-script bevat. Door app te initialiseren met Dash, leggen we de basis voor het opbouwen van een interactieve webapplicatie waarin we gebruik kunnen maken van alle functionaliteiten van Dash, inclusief interactieve datavisualisaties met behulp van Plotly.

### App starten

Om de Dash-applicatie te laten draaien op een lokale server, maken we gebruik van een specifieke Python-conventie. We beginnen met het controleren van de speciale variabele \_\_name\_\_. Deze variabele bevat de naam van het huidige module of script dat wordt uitgevoerd. Als \_\_name\_\_ gelijk is aan \_\_main\_\_, betekent dit dat het script direct wordt uitgevoerd als het hoofdprogramma.

In ons geval gebruiken we deze conditie om ervoor te zorgen dat de Dash-applicatie wordt gestart wanneer het script rechtstreeks wordt uitgevoerd. Dit doen we door het app.run\_server(debug=True) commando uit te voeren. Hierbij verwijst app naar het Dash-applicatie-object dat we eerder hebben geconfigureerd en opgebouwd.

De parameter debug=True die we doorgeven aan run\_server() activeert de debug-modus van Dash. Deze modus zorgt ervoor dat er gedetailleerde foutmeldingen en nuttige informatie worden weergegeven in de console tijdens het draaien van de applicatie. Dit is vooral handig tijdens het ontwikkelen en testen van de Dash-applicatie.

Samengevat, door deze conventie en configuratie te gebruiken, kunnen we onze Dash-applicatie effectief laten draaien op een lokale server. Dit stelt ons in staat om interactieve webapplicaties te ontwikkelen met geavanceerde datavisualisaties en interactieve gebruikerscomponenten, zoals grafieken en dropdowns, binnen een Python-omgeving.

Afbeelding met tekst, Lettertype, schermopname, klok

Automatisch gegenereerde beschrijving