|  |  |
| --- | --- |
| Biostatica  Door Mark Schrauwen en Alistair Vardy | Matlab Wk4.2 |

Inhoudsopgave

[Versiebeheer 2](#_Toc487553312)

[1 Inleiding 3](#_Toc487553313)

[2 Schrijven naar een bestand 4](#_Toc487553314)

[2.1 Delimiters 5](#_Toc487553315)

[2.2 Comma Separated Values (CSV-bestanden) 5](#_Toc487553316)

[2.2.1 De extensie van een CSV-bestand 6](#_Toc487553317)

[2.2.2 Het lezen van CSV-bestanden 7](#_Toc487553318)

[2.2.3 Absolute en relatieve paden 7](#_Toc487553319)

[2.3 Oplossing: wegschrijven van kolomvectoren 8](#_Toc487553320)

[2.4 Vragen en opdrachten 9](#_Toc487553321)

[2.5 Antwoorden 10](#_Toc487553322)

[3 Het lezen van een bestand 11](#_Toc487553323)

[3.1 Headers 11](#_Toc487553324)

[3.1.1 Opbouw van de volgende secties 12](#_Toc487553325)

[3.2 Het inlezen van eenvoudige tekstbestanden 12](#_Toc487553326)

[3.3 Wat te doen als het inlezen van een bestand niet werkt? 14](#_Toc487553327)

[3.3.1 Inlezen van bestanden m.b.v. automatische Matlab dialoog 14](#_Toc487553328)

[3.4 MAT-bestanden (opslaan en laden van Workspace) 17](#_Toc487553329)

[3.5 Vragen en opdrachten 18](#_Toc487553330)

[3.6 Antwoorden 19](#_Toc487553331)

[4 Excel bestanden 20](#_Toc487553332)

[4.1 Vragen en opdrachten 20](#_Toc487553333)

[4.2 Resultaten 21](#_Toc487553334)

# Versiebeheer

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Datum | Beschrijving | Door |
| 0.0 | 11-07-2017 | Eerste versie | Mark Schrauwen |
|  |  |  |  |
|  |  |  |  |

# Inleiding

Begin deze week is je Matlab kennis verder uitgebreid met functies die je helpen om XXXX. Deze kennis is belangrijk voor de blokken die nog komen gaan en natuurlijk de hoofdfase. Ook heb je wat ervaring opgedaan met het lezen van de help/doc bestanden en ben je nu beter in staat om de soms wat abstracte beschrijvingen te lezen en te begrijpen.

In dit laatste deel van deze Matlab cursus ga je bestanden lezen en schrijven. Deze activiteiten komen heel vaak voor en zijn bijna altijd anders. Het (in)lezen van data van meting1 is vaak heel anders dan het inlezen van data van meting 2.

Ook zal je vaak data moeten wegschrijven, bijvoorbeeld de data van de output van een data model wat je met Matlab hebt gemaakt. Dat leer je in deze reader. Één manier om de data wegschrijven is met behulp van een .mat bestand. Deze bestanden zijn specifieke Matlab bestanden en het bestandsformaat is ontwikkeld door Matlab. Ter vergelijking, het lezen en schrijven van bestanden zijn gemaakt met een algemene extensie (namelijk een .txt extensie).

De afbeeldingen in dit deel van de reader laten steeds vaker zien hoe de hele Matlab omgeving er uit ziet. Probeer jezelf aan te leren elke regel code, dus ook het commentaar in het groen, door te lezen en te begrijpen. Je zult je dus moeten focussen op de code en niet op de omgeving (zoals de Current Folder, Workspace, etc.).

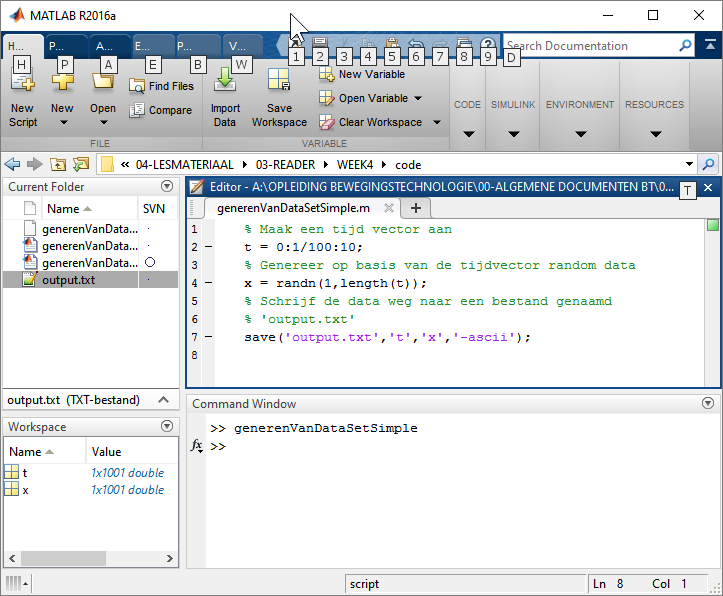
Ook deze week komt er nog een eindopdracht. Je zult in deze opdracht veel van de eerder opgedane kennis nodig hebben.

Volgende week krijg je een (digitale) toets. Deze toets bestaat uit alternatieve versies van de weekopdrachten. De moeilijkheidsgraad van het tentamen is dus zeer vergelijkbaar met de moeilijkheidsgraad van de week eindopdrachten.

**Zie je een fout? Of heb je een aanbeveling dan horen we dat graag! Stuur dan een e-mail naar** [**mjschrau@hhs.nl**](mailto:mjschrau@hhs.nl) **en wij passen het dan z.s.m. aan.**

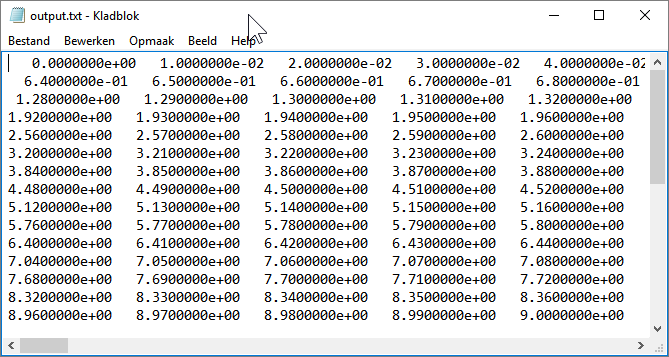
# Schrijven naar een bestand

Als een Bewegingstechnoloog data heeft verkregen van een ADC of een andere meetapparaat dan wordt deze data vaak aangeleverd vanuit een .txt, .dat of een .asc bestand. Dat zijn extensies voor tekstbestanden (die je met een texteditor) kunt lezen.



Figuur 1: Een code voorbeeld waarin data is gegeneerd in het bestand ‘output.txt’

Het resultaat de code in Figuur 1 is hieronder te zien:



Figuur 2: De output van de code in Figuur 1.

Laten we eens onderzoeken wat hier is gebeurd. In Figuur 2 is het resultaat te zien van de code in Figuur 1. Hier gaat echter iets mis. Waar is bijvoorbeeld de tijd-as te zien? Als je bekijkt wat er in de aangemaakte variabele **t** staat valt direct op wat er hier gebeurt.

Maak een nieuwe m-file aan in een logische benoemde folder. Schrijf de code van Figuur 1 over en sla de m-file op. Voer de m-file uit.

Controleer wat er in de variabele t staat

Wat valt op?

De tijdvector (variabele **t**) bestaat uit kolommen in plaats van rijen (een rijvector in plaats van een kolomvector). Normaliter vinden we het prettiger om data per vector in een kolom te hebben staan (een kolomvector). Dat leest namelijk prettiger. Het maakt Matlab niet uit of je met kolomdata of rijdata werkt, als je het maar zorgvuldig doet.

Pas de code aan zodat de data wordt gegeneerd in kolomvectoren en sla de data op in een bestand genaamd ‘kolomvectoroutput.txt’. Tip: maak van de kolomvectoren een matrix en exporteer de matrix. De oplossing van deze opdracht staat in paragraaf 2.3.

Wat zijn de verschillen tussen de code van voorgaande opdracht en de code in Figuur 1? Schrijf ze hieronder op (of als je geheel digitaal werkt, schrijf ze op in een apart bestand):

*In de code staat deze regel:* save('outputkolom.txt','data','-ascii');

is ‘-ascii’ nodig?

Test dit uit! Tip: verwijder eerst het bestand dat je gaat wegschrijven.

Als je het voorgaande hebt getest, haal dan ‘.txt’ weg uit de string: ‘outputkolom.txt’ en kijk wat er na het uitvoeren dan gebeurd.

## Separators

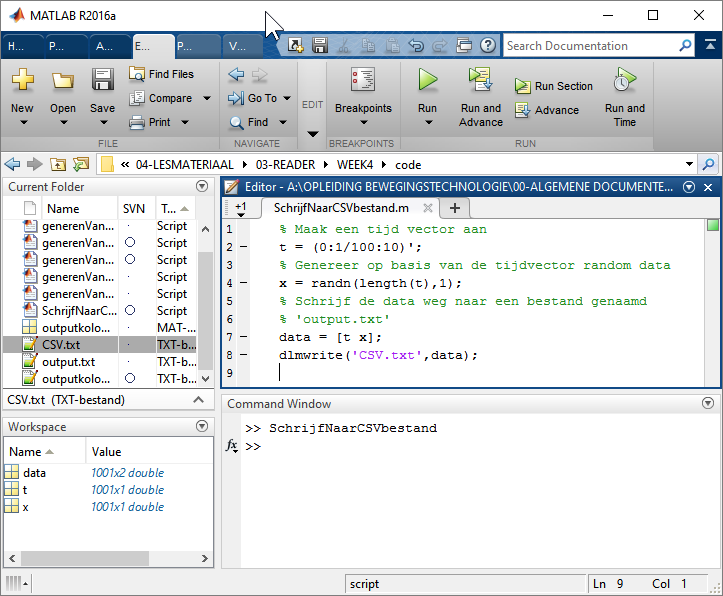
*Wat betekent de term ‘separator?*

De term separator (soms ook ‘delimiter’ genoemd) kom je met regelmaat tegen als je data wegschrijft of data probeert te lezen. Maar wat is een separator precies? Een separator is het symbool die je gebruikt om opeenvolgende datawaardes te scheiden van elkaar. Het resultaat van de code in Figuur 1 is dat de data gescheiden wordt door de spatie. Het spatie teken is dan een separator. Standaard gebruikt Matlab dus een spatie als separator. Maar er zijn ook andere separators mogelijk.

## Comma Separated Values (CSV-bestanden)

Een veel voorkomend bestandformaat waarin data wordt geëxporteerd, zijn CSV-bestanden. Deze bestanden worden ook wel komma-gescheiden-bestanden genoemd. De separator bestaat hier uit een komma in plaats uit een spatie of een andere symbool.

Het schrijven naar CSV-bestanden doen we met de dlmwrite() functie die standaard aanwezig is in Matlab:



Figuur 3: het opslaan van data in een CSV-bestand.

**Probeer zelf een CSV-bestand weg te schrijven op basis van de code in Figuur 3.**

**Controleer of dit is gelukt door het bestand te openen met Microsoft Excel.**

*Zitten er spaties in een CSV-bestand?*

### De extensie van een CSV-bestand

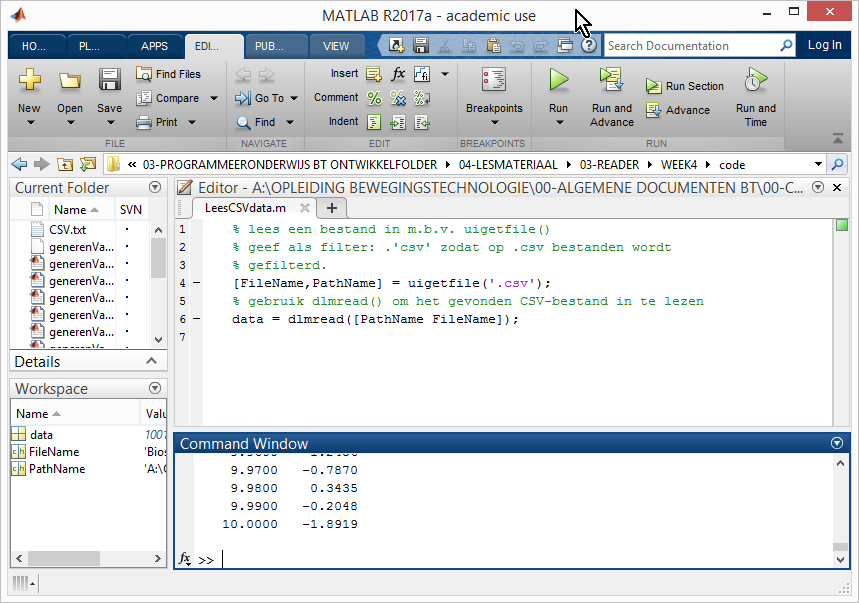
Een CSV-bestand heeft vaak de extensie ‘.csv’. Echter is de opmaak nog steeds een tekst-bestand (en het kan dus een ‘.txt’ extensie hebben). Je kunt de code Figuur 3eenvoudig aanpassen zodat in plaats van .txt bestanden .csv bestanden worden weggeschreven.

*Welke extensie heeft de voorkeur bij een CSV-bestand? Een .txt extensie of een .csv extensie?*

De extensie van een willekeurig bestand geeft aan met wat voor bestand je te maken hebt. Het ligt voor de hand om bij een CSV-bestand automatisch een csv-extensie te gebruiken. Dit is niet verplicht maar wel zo duidelijk.

### Het lezen van CSV-bestanden

Het schrijven van CSV-bestanden doen we met dlmwrite() en is niet veel moeilijker dan het wegschrijven van data m.b.v. save(). Het lezen van CSV-bestanden gaat op vergelijkbare wijze. Voor het inlezen van CSV-bestanden gebruik je de functie dlmread():



Figuur 4: het lezen van een CSV-bestand m.b.v. de functie dlmread().

In Figuur 4 is te zien hoe een CSV-bestand ingelezen kan worden. Er gebeuren een aantal dingen die je nog niet eerder hebt gezien. Zo zie je in regel 4 in Figuur 4 een nieuw stuk code.

*Wat doet deze regel code?*

De functie uigetfile() opent een *open-bestand-dialoog*. Als je een bestand moet inlezen dan moet je weten welk bestand je gaat inlezen. Daarom moet eerst een bestand geselecteerd worden en dat doe je met de Matlab functie uigetfile(). De output van uigetfile() hangt van de acties die jij als gebruiker uitvoert. Als je bijvoorbeeld dit doet: bestandnaam = uigetfile(); dan geeft uigetfile() alleen een bestandnaam terug en niet het pad waar het bestand zich op bevind.

De functie geeft drie elementen terug: een FileName (bestandnaam) en een PathName. De variable PathName geeft het volledige absolute path van het aangeklikte bestand.

**Typ in het Command Window: help uigetfile()**

*Wat voor reactie geeft deze functie als je een bestand selecteert dat fout is of niet bestaat?*

### Absolute en relatieve paden

*Deze subparagraaf mag worden overgeslagen daar het niet behoort tot de leerstof*

*Hoe geef je aan waar een bestand zich bevind op je computer?*

De locatie van een bestand kun je op twee manieren opgeven: met een absoluut pad of een relatief pad. Meestal gebruik je een absoluut als je zelf je scripts ontwikkeld maar relatieve paden kunnen gemakkelijk zijn.

*Wat is het verschil tussen een relatief pad en een absoluut pad?*

Een absoluut pad is een pad waar je expliciet opgeeft in welke (sub)folders een bestand staat:

C:/Program Files/Matlab/Tekstbestanden/tekst.txt

Een relatief pad geef je op ten opzichte van het bestand waarin je het relatieve pad aanroept. Stel voor, je werkt vanuit een bestand genaamd openEenBestandMbvRelatiefPad.m. Dat bestand staat in de absolute locatie:

C:/temp/matlabprobeersel/

Je wilt met openEenBestandMbvRelatiefPad.m een tekstbestand openen dat in de volgende folder staat:

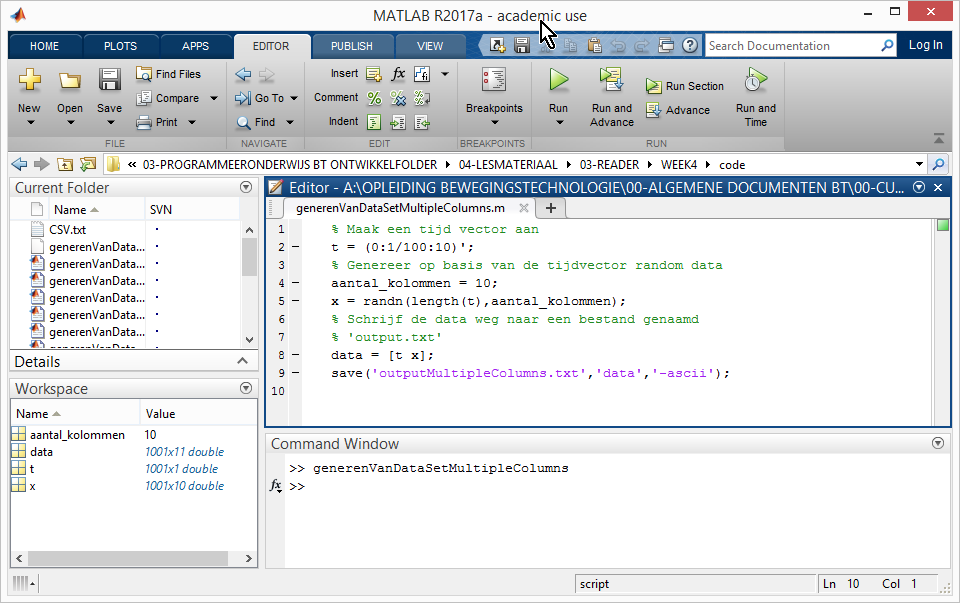
C:/temp/data.csv

Dan kun je dat bestand openen, vanuit openEenBestandMbvRelatiefPad.m, m.b.v. het relatieve path:

../data.csv.

Hier betekent ../ dat je één folder omhoog vanuit openEenBestandMbvRelatiefPad.m. Omdat openEenBestandMbvRelatiefPad.m in de folder C:/temp/matlabprobeersel/ staat is de hoger liggende folder C:/temp/ en in die folder staat nou net het bestand data.csv.

## Oplossing: wegschrijven van kolomvectoren



Figuur 5: het wegschrijven van kolomvectoren in plaats van rijvectoren.

## Vragen en opdrachten

1. Wat zijn de verschillen tussen de code in Figuur 1 en de opdracht waarin jij de data moest exporteren in kolomvectoren?
2. *In de code staat deze regel:* save('outputkolom.txt','data','-ascii');

*is ‘-ascii’ nodig?*

1. Wat gebeurt er als je ‘-ascii’ weglaat en je haalt ‘.txt’ weg?
2. Zitten er spaties in een CSV-bestand?
3. Waar gebruik je de functie uigetfile() voor?
4. [Lees dit CSV-bestand in met Matlab](https://www.dropbox.com/s/j9csyhyg6lvy9iq/Biostatica_Voorbeeld_CSV_data.csv?dl=1). Wat zijn de dimensies (afmetingen) van de variabele waarin de ingelezen CSV-data staat?
5. Welke regel moet je aanpassen in de code van Figuur 3
6. Noem een aantal in dit document genoemde bestand extensies van tekstbestanden.
7. In Figuur 4 wordt de output van de functie uigetfile() in twee variabele opgeslagen. De variabele zijn FileName en PathName. Mag je deze variabelen ook een andere naam geven? Waarom is dat?

## Antwoorden

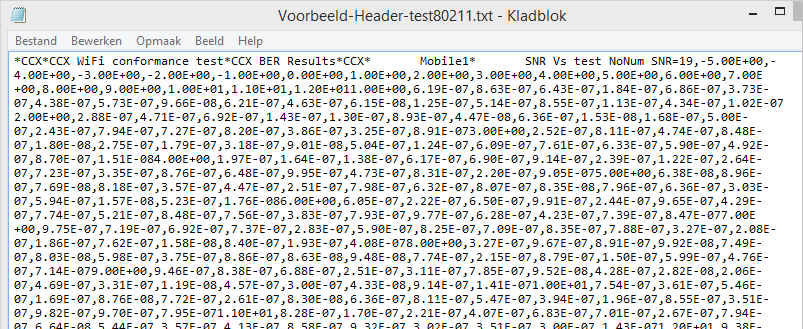
1. De aangemaakte tijdvector is een kolomvector geworden. De random gegeneerde data is ook een kolomvector geworden. Dit gebeurt echter niet vanzelf. Je moet de parameters (1, length(t)) namelijk omdraaien. Als je de tip hebt opgevolgd dat heb je de tijdvector en de datavector ook in een matrix gezet.
2. Ja, dit is echt nodig. Open het bestand maar eens. Daarin staat leesbare data als je ‘-ascci’ hebt weggelaten in de regel. Je moet dus expliciet opgeven dat je tekstdata wegschrijft als je de functie save() gebruikt in Matlab.
3. Dan slaat Matlab de data automatisch op in een .mat bestand (dat is een bestandsformaat van Matlab).
4. Nee, dat is als de komma gescheiden data zelf geen spaties bevat.
5. Dat is een standaard ingebakken Matlab functie die de programmeur in staat stelt om vragen te stellen aan de gebruiker van de code.
6. 1001x2 double
7. Regel 8, schrijf daar in plaats van .txt .csv en je bent klaar. **Probeer dit!**
8. .txt, .dat, .asc, .csv
9. Het betreft hier variabelen. Een variabele mag je zelf een naam geven. Je mag dus de namen veranderen. Het is de kunst om een variabelenaam zo helder mogelijk te houden zodat iedereen in één keer kan zien wat de variabele bevat.

# Het lezen van een bestand

Het lezen van bestanden is lastiger dan het schrijven naar een bestand. Dat verwacht je in eerste instantie niet. In het vorige hoofdstuk heb je een CSV-bestand weggeschreven en ingelezen en dat ging zonder al te veel moeite. Het voordeel van een CSV-bestand is dat het een normaal tekstbestand is maar met de extra afspraak dat alle data gescheiden is door een komma. Deze afspraak maakt het al gemakkelijker om data in te lezen. Er zijn nog meer afspraken denkbaar om een data bestand beter te kunnen interpreteren.

## Headers

Een veel voorkomende manier om ruwe data beter te interpreteren is om een *header* aan een data-bestand toe te voegen. Een header van een data-bestand zijn één of meerdere regels die extra informatie geven over de data en/of het bestand. In Figuur 6 is een voorbeeld te zien van een bestand met een header.



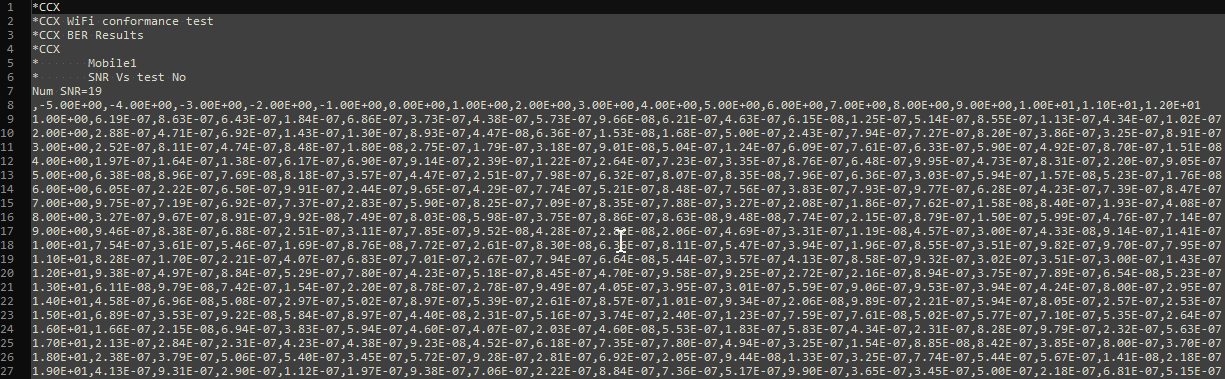
Figuur 6: Data met een header. Dit tekstbestand is een CSV-bestand (zie de komma’s) maar dan met een header.

[**Download de data van Figuur 6 hier**](https://www.dropbox.com/s/nmbw3nsfb5rf49u/Voorbeeld-Header-test80211.txt?dl=1)

**Lees de data van Figuur 6 in m.b.v. de code in Figuur 4**

Na het uitvoeren van de voorgaande acties zie je dat de code van Figuur 4 plotseling niet meer werkt. Dat komt omdat in dit bestand een header staat.

|  |
| --- |
| **De data in Figuur 6 is geopend met behulp van Notepad. Echter houdt Notepad niet altijd rekening met de indeling van een bestand. De volgende afbeelding houdt hier wel rekening mee.** |



Figuur 7: hetzelfde bestand als in Figuur 6 maar dan ingelezen met een applicatie (Notepad++) die rekening houdt met de indeling van een bestand.

Je merkt op dat het niet gemakkelijk is om een bestand in te lezen. Je moet namelijk rekening houden met hoe informatie is ingedeeld. Zodra een bestand eenvoudig is ingedeeld (bijvoorbeeld alleen maar datakolommen heeft) is inlezen gemakkelijker. Maar bij één extra regel hoeft bestaande code niet meer te werken.

### Opbouw van de volgende secties

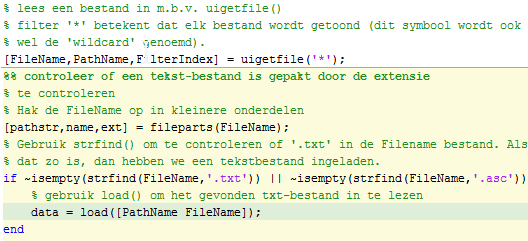
In de volgende sectie beginnen we met het inlezen van een eenvoudig bestand. Een bestand zonder headers en andere toevoegingen. Daarna gaan we wat andere bestanden inlezen. Uiteindelijk ga je zelf een aantal oefeningen doen die je ervaring geven met het inlezen van verschillende bestanden. De vaardigheden die je hier leert, zal je met regelmaat in de praktijk moeten gebruiken.

Er moet worden opgemerkt dat er in Matlab tegenwoordig functionaliteit is ingebouwd die automatisch een bestand, ongeacht de opmaak, probeert in te lezen. Dit heeft een groot voordeel: het wordt namelijk gemakkelijker om bestanden in te lezen. Het heeft ook een nadeel. De automatisch gegeneerde code is veel minder efficiënt. De auteurs van deze reader hebben de ervaring gehad dat deze automatisch vorm van inlezen soms 10x zo lang duurt dan een zelfgeschreven variant. Voor de volledigheid laten we toch zien hoe je m.b.v. Matlab een bestand kunt inlezen.

## Het inlezen van eenvoudige tekstbestanden

In dit gedeelte gaan we eenvoudige tekst-bestanden (dus niet CSV-bestanden per se) inlezen. We nemen aan dat een tekstbestand de extensie .txt of .asc heeft. Als dat niet het geval is, nemen we aan dat het geen tekst-bestand betreft.

De code in Figuur 8 laat zien hoe je een tekstbestand inleest:



Figuur 8: de code om een simpel tekstbestand in te lezen.

**Lees elke regel van de code in Figuur 8 door en lees (altijd) het commentaar**

Je bent inmiddels bekend met de functie uigetfile(). Na het uitvoeren van uigetfile() moet worden gecontroleerd het juiste bestand is ingelezen. Daarvoor gebruiken we wat digitale logica (zie de NOT-operator en de OR-operator). De logica is er om meerdere testen uit te kunnen voeren, namelijk is er een .txt of een .asc bestand ingelezen. Het testen of het juiste bestand is ingelezen wordt gedaan m.b.v. de functie strfind(). Deze functie zoekt naar een bepaalde string in een bepaalde string. Dus als je de tekst: “Mark en Alistair ontwerpen een Matlab cursus” hebt kun je met behulp van strfind() kijken of het woord “ontwerpen” in de string staat (wat zo is). Deze functie geeft vervolgens terug op welke plek de specifieke string is gevonden. Als er niets is gevonden dan geeft de functie strfind() een lege vector terug. Met behulp van de functie isempty() kan worden gecontroleerd of een vector leeg is of niet. In dit geval mag alleen een bestand worden ingelezen als het de juiste extensie heeft. Dan moet die extensie ook zijn gevonden en mag de output van strfind() niet leeg zijn. Daarom staat de NOT-operator voor de isempty() functie zodat de if-statement op het juiste moment wordt uitgevoerd.

**Gebruik de code in Figuur 8 om** [**dit bestand**](bewegingstechnologie.com/weblinks/curr17/outputMultipleColumns) **in te lezen.**

**Download nu** [**dit bestand**](http://www.bewegingstechnologie.com/weblinks/curr17/outputMultipleColumnsHeader) **en vergelijk het met het vorige bestand**

**Gebruik de code in Figuur 8 om dat bestand in te lezen, wat valt op?**

In de laatste stap is het inlezen mislukt. Waarom? Er staat een korte header in het tekstbestand en daar kan load() niet mee omgaan.

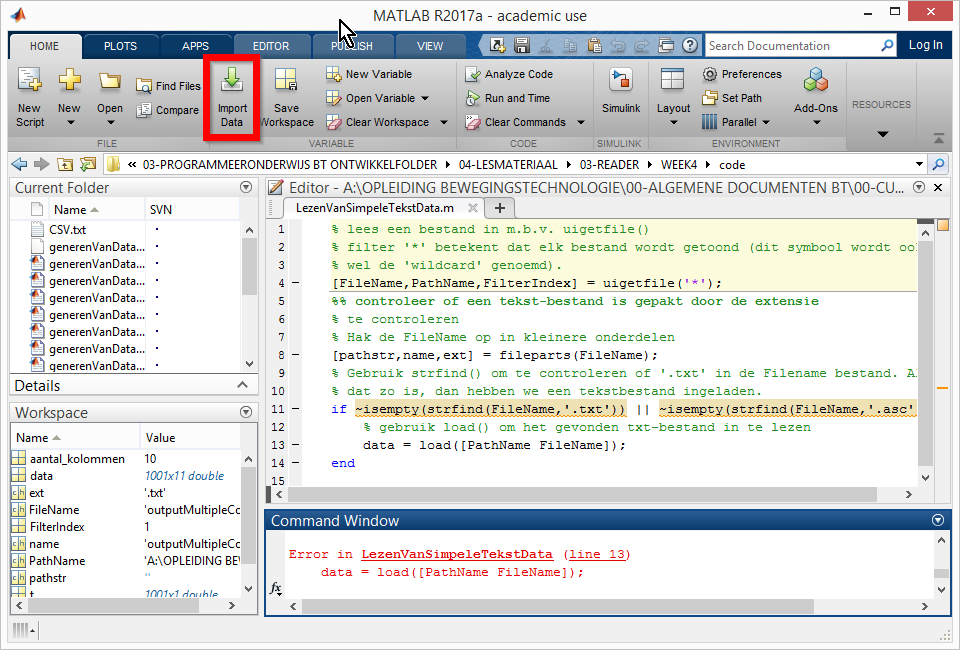
## Wat te doen als het inlezen van een bestand niet werkt?

Er zijn een aantal stappen die je kunt doorlopen om het inlezen van een bestand in Matlab werkend te krijgen. Welke van de genoemde stappen het beste is, ligt aan de applicatie en de wensen van de gebruiker. In de volgende, niet uitputtende lijst, staan een aantal veel voorkomende oplossingen als een bestand niet kan worden ingelezen i.v.m. een foutmelding. We nemen hieronder aan dat het om bestanden gaat met leesbare karakters (lees: dus niet binaire bestanden):

* Haal de header van het tekstbestand weg. Indien er geen tot weinig relevante informatie staat in de header, haal deze dan weg.
* Controleer het data-bestand op Chinese of bijzondere karakters en haal deze weg.
* Is het een specifiek formaat dat je probeert in te lezen? Kijk dan eens op een van de vele Matlab fora. Vaak hebben mensen jouw probleem al opgelost.
* Maak gebruik van de ingebouwde Matlab functionaliteit die het inlezen van specifieke bestanden voor jou automatiseert.

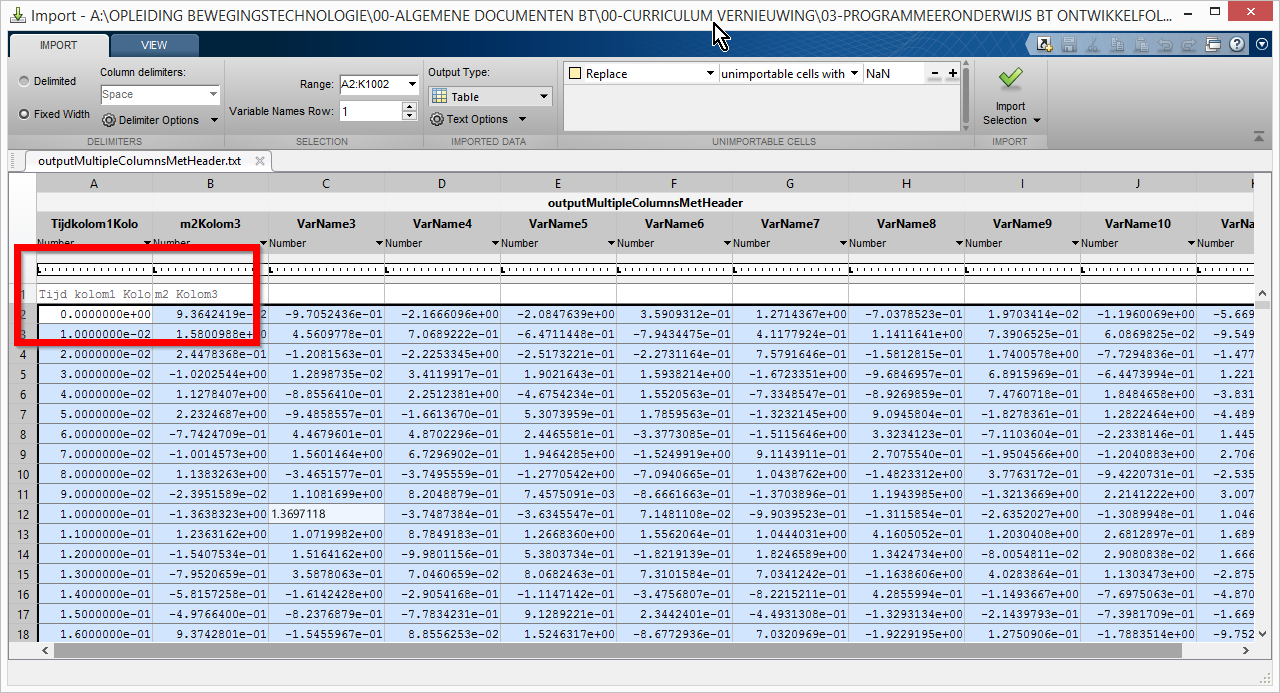
### Inlezen van bestanden m.b.v. automatische Matlab dialoog

In Figuur 9 is aangegeven welke knop je kunt gebruiken om een Matlab dialoog te starten:



Figuur 9: het importeren van data met behulp van Matlab dialoog

**Klik op deze button en kies het eerder gedownloade bestand (**[**dit bestand**](http://www.bewegingstechnologie.com/weblinks/curr17/outputMultipleColumnsHeader)**).**

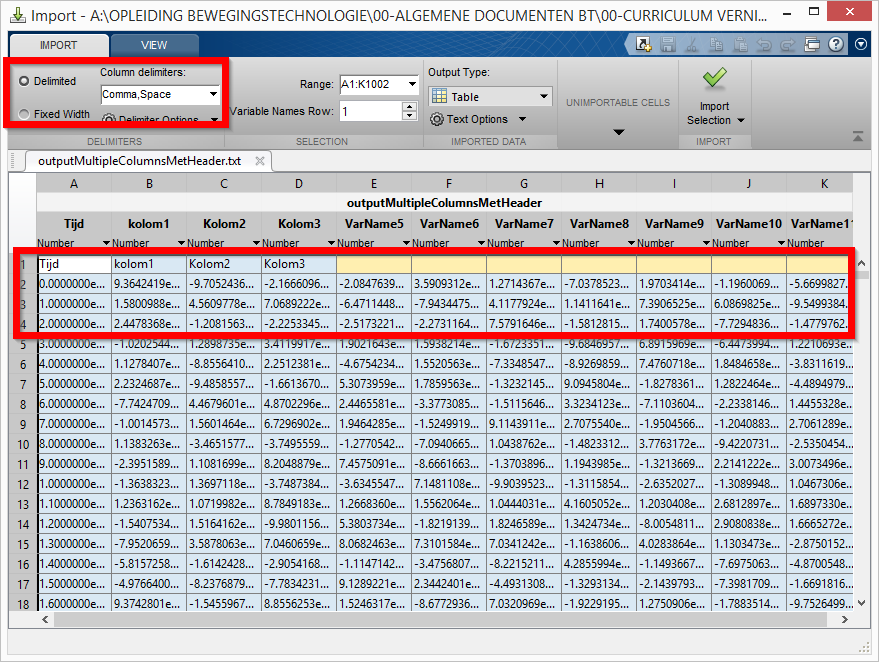


Figuur 10: dit is het venster dat je krijgt te zien. Wat zie je hier?

Na het uitvoeren van de vorige actie krijg je het venster in Figuur 10 te zien. Je ziet in Figuur 10 dat de data in het tekstbestand al is ingelezen. Echter is het nog niet helemaal goed ingelezen. Het rode vierkant laat dat zien. De header gegevens zijn in licht grijs weergegeven. De data in de header hoort echter bij specifieke kolommen. De eerste 4 kolommen hebben namelijk een naam gekregen.

Hoe lossen we dit op?

Je zult nog wat instellingen moeten veranderen, namelijk deze instellingen:



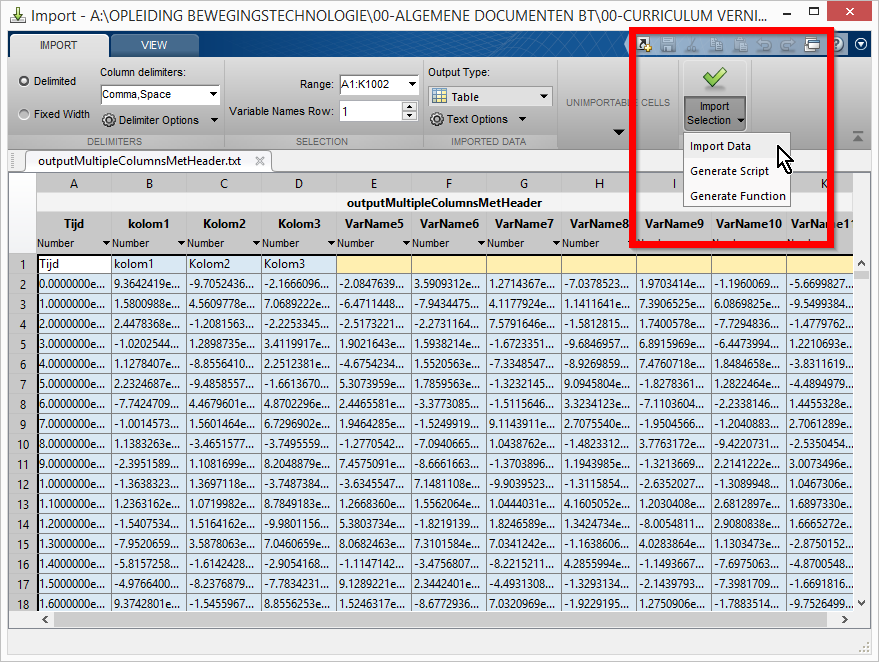
Figuur 11: met de juiste instellingen (zie Delimited, linker vierkant) wordt de data correct weergegeven.

Zie je welke instellingen zijn veranderd?

In Figuur 11 zijn de juiste separators (separators) gekozen zodat Matlab weet welke separators moeten worden gebruikt. Het betreft een CSV-bestand (met een tekst extensie) en de header is gescheiden door spaties. Daarom heb je de separators spatie en komma opgegeven.

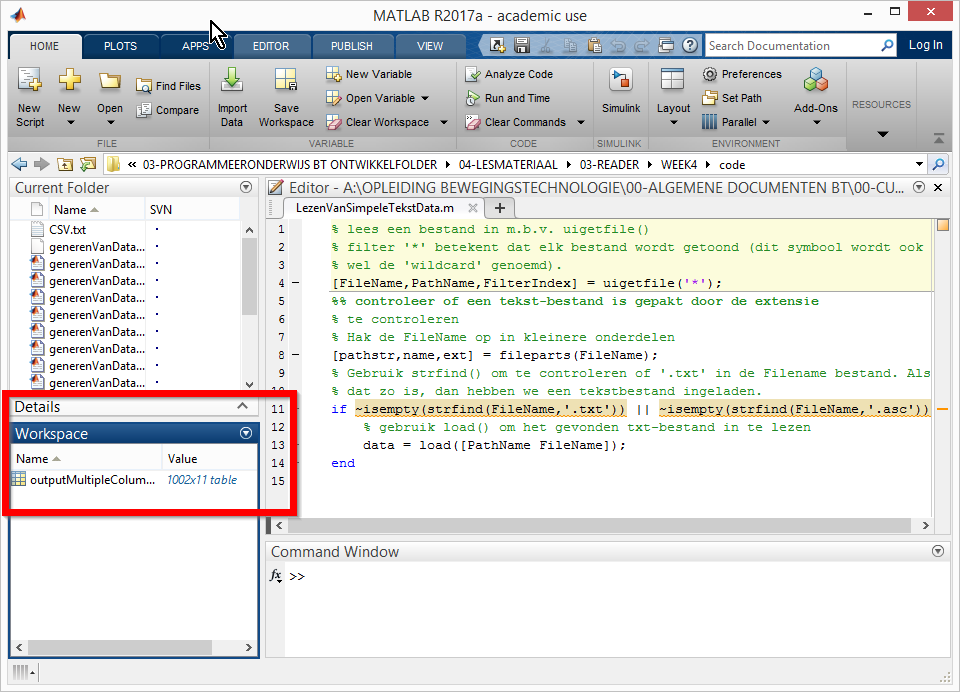
Dan komt nu het handige. Je kunt in de volgende stap (Figuur 12) kiezen wat je met de data wilt gaan doen. Je kunt kiezen uit:

1. Import Data: de data wordt ingelezen (verder niets)
2. Generate Script: er wordt een script gemaakt dat vergelijkbare bestanden kan inlezen
3. Generate Function: er wordt een functie gemaakt dat vergelijkbare bestanden kan inlezen



Figuur 12: het inlezen van data afronden.

Alleen optie 1 wordt hier besproken. De overige opties kan de lezer snel en gemakkelijk zelf testen en proberen. Bij de optie Import Data wordt de ingelezen data in een Matlab Table geplaats



Figuur 13: resultaat van het inlezen m.b.v. de Matlab dialoog.

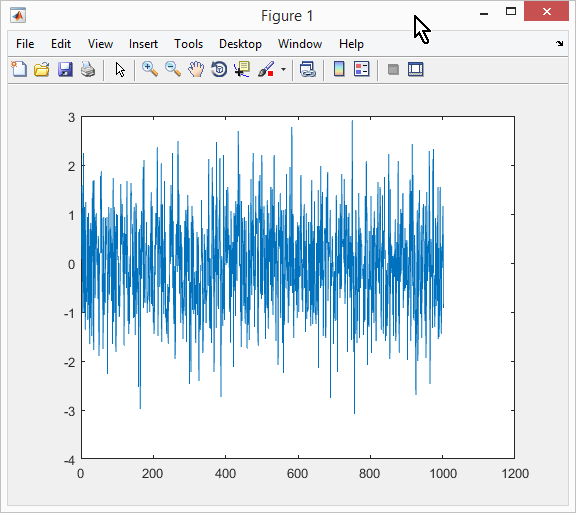
**Plot de data in de eerste kolom**

Je bent er zojuist achter gekomen dat het plotten van de data in de Matlab Table niet goed gaat. Dat komt omdat de data in een bestandsformaat staat dat niet kan worden geplot.

*Wat is de oplossing?*

Je moet de Matlab Table (outputMultipleColumnsMetHeader) omzetten naar een Matrix en dat doe je met de functie: table2array().

**Probeer nogmaals de data in de eerste kolom te plotten het resultaat staat in**

****

Figuur 14: Dit moet het resultaat zijn van het inlezen

## MAT-bestanden (opslaan en laden van Workspace)

Matlab heeft een eigen bestandsformaat met de extensie .mat. In sommige gevallen is het handig om je data op te slaan in MAT-bestanden. Je kunt je hele Workspace opslaan in een MAT-bestand om het later weer te kunnen gebruiken. Om een MAT-bestand te genereren van de Workspace doe je: save(‘woef.mat’). Om woef.mat in te lezen gebruik je het commando load(‘woef.mat’).

**Sla jouw huidige Workspace op.**

**Start Matlab opnieuw op**

**Laad het eerder opgeslagen .mat bestand opnieuw in.**

## Vragen en opdrachten

1. Wat is het nadeel van het door Matlab semi-automatisch laten inlezen van een data bestand met een specifieke opmaak en/of header?
2. [Lees dit bestand](http://www.bewegingstechnologie.com/weblinks/curr17/voorbeelddata/test23122016Cal01.txt) in zonder de **Import Data** knop.
3. Lees het voorgaande bestand in m.b.v. de Import Data knop
4. Welke extensie heeft het bestandsformaat van Matlab bestanden?
5. Wanneer is het opslaan van data in het Matlab formaat in de praktijk handig?

## Antwoorden

1. Dat het niet altijd goed gaat. Bijvoorbeeld sommige bestanden hebben halverwege een datakolom ook een header regel staan. Bij lange bestanden kan het inlezen heel lang duren en is het de moeite waard om zelf een inlees-script te schrijven.
2. Dit is een opdracht
3. Dit is een opdracht
4. .Mat
5. Als je snel alle variabelen en vectoren die in het .mat bestand weer nodig hebt voor een volgende sessie.

# Excel bestanden

Matlab heeft het inlezen en het schrijven naar Excel bestanden gemakkelijk gemaakt. Er zijn maar een paar functies die je moet kennen om moeiteloos naar Excel te schrijven en Excel bestanden te lezen.

In een praktijk situatie waar jij zelf moet bedenken hoe je een Matlab probleem/situatie/uitdaging moet oplossen zal je vaak op zoek moeten naar de juiste oplossing. Je ontkomt dan niet aan het lezen van de Matlab Documentation.

In plaats van een volledige beschrijving van de uit te voeren stappen en de potentiele problemen geven we hieronder een aantal opdrachten en de resultaten van de uitwerkingen. De tussenstappen worden niet gegeven en moet je zelf bedenken. Hoe? Door de Matlab Documentation door te lezen. Om het je gemakkelijk te maken:

1. Lees eerst de onderstaande vragen/opdrachten door
2. Ga dan naar: <http://nl.mathworks.com/help/matlab/ref/xlsread.html>
3. Haal de benodigde informatie uit de documentatie en schrijf de code die je moet genereren voor de opdrachten

## Vragen en opdrachten

1. [Download dit Excelbestand](http://www.bewegingstechnologie.com/weblinks/curr17/voorbeelddata/Kinec-tics-20170511120912_angles.xlsx). Open het en onderzoek het bestand m.b.v. Excel. Probeer daarna m.b.v. Matlab van het eerste tabblad de eerste 6 kolommen te plotten.
2. [Download dit Excelbestand](http://www.bewegingstechnologie.com/weblinks/curr17/voorbeelddata/ie_data.xls). Plot de data in kolom D en H van het tabblad ‘Data’.
3. Genereer een sinus, cosinus en een tangens op basis van vector t=0:1/10:100; met amplitude 2. Schrijf deze signalen naar afzonderlijke kolommen en in het tweede tabblad.

## Resultaten

1. 
2. 
3. 