|  |  |
| --- | --- |
| Biostatica  Door Bart van Trigt, Alistair Vardy en Mark Schrauwen | Matlab Wk2.1 |

Inhoudsopgave

[Versiebeheer 2](#_Toc494730387)

[1 Inleiding 3](#_Toc494730388)

[2 Herhaling functies vorige week 4](#_Toc494730389)

[3 Functies met meerdere outputs 5](#_Toc494730390)

[3.1 Vragen en opdrachten 5](#_Toc494730391)

[3.2 Antwoorden en uitwerkingen 6](#_Toc494730392)

[4 Functie met meerdere inputs 7](#_Toc494730393)

[4.1 Vragen en opdrachten 10](#_Toc494730394)

[4.2 Antwoorden en uitwerkingen 10](#_Toc494730395)

[5 Ingewikkeldere functies maken 11](#_Toc494730396)

[5.1 Vragen en opdrachten 13](#_Toc494730397)

[5.2 Antwoorden en uitwerkingen 13](#_Toc494730398)

[6 Specifieke terminologie en eigenschappen 14](#_Toc494730399)

[7 Handige Matlab functies 16](#_Toc494730400)

[7.1 Vragen en opdrachten 18](#_Toc494730401)

[7.2 Antwoorden en uitwerkingen 19](#_Toc494730402)

# Versiebeheer

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Datum | Beschrijving | Door |
| 0.0 | 11-07-2017 | Eerste versie | Alistair Vardy |
| 0.1 | 13-09-2017 | Under construction | Bart van Trigt |
| 0.2 | 14-09-2017 | Meerder outputs en meerdere inputs klaar  Nu verder met ingewikkeldere functies | Bart van Trigt |
| 0.3 | 18-09-2017 | Ingewikkeldere functies | Bart van Trigt |
| 0.4 | 19-09-2017 | Afronding versie 1 | Bart van Trigt |
| 0.5 | 02-10-2017 | Paar opmerkingen van Chadier Wilson en Denice Vis verwerkt. | Mark Schrauwen |
|  |  |  |  |

# Inleiding

We hebben vorige week heb je kennis gemaakt met MATLAB. Als het goed is weet je nu waarom jij als bewegingstechnoloog MATLAB goed moet gebruiken. Vorige week heb je geleerd wat operatoren zijn, hoe je kan debuggen, hoe je vectoren maakt in MATLAB, standaard functies in MATLAB en je hebt al een keer een functie zelf gemaakt! We gaan in deze reader verder met het maken van een functie. Tot nu toe heb je makkelijke functies gemaakt, je gaf een input en er kwam een output uit. Maar wat nu als we bijvoorbeeld een formule hebben die meerder inputs heeft.

De volgende leerdoelen zijn hiervoor gemaakt, na het doorwerken van deze reader kan de student:

1. In eigenwoorden uitleggen wat een functie in Matlab is.
2. Gebruik maken van standaard functies in Matlab.
3. Een functie met meerdere outputs aanroepen.
4. Een functie met meerdere inputs aanroepen.
5. Zelf een functie aanmaken in Matlab met meerdere inputs en outputs.
6. Uitleggen dat een matrix rijen en kolommen bevat.
7. In eigenwoorden inleidende Matlab terminologie uitleggen, zoals: colon, operator, indices, etc.
8. De help functie raadplegen en uitleggen waarom deze belangrijk is bij het gebruik van Matlab.

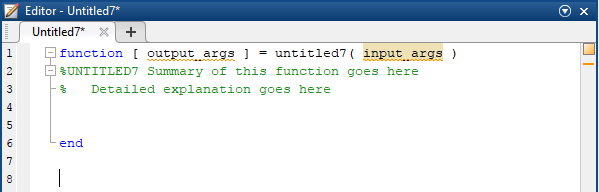
**Zie je een fout? Of heb je een aanbeveling dan horen we dat graag! Stuur dan een e-mail naar** [**mjschrau@hhs.nl**](mailto:mjschrau@hhs.nl) **en wij passen het dan z.s.m. aan.**

# Herhaling functies vorige week

Vorige week heb je kennis gemaakt met functies. Deze reader gaat hier verder op in. Om deze reader goed te begrijpen is het van belang dat je de reader van vorige week hebt afgerond. Mocht je dit niet gedaan hebben zorg ervoor dat je dit eerst afmaakt (reader 1\_2, hoofdstuk 4).

Een korte herhaling:

* Je kan een functie aanroepen in je Command Window, door zijn functienaam te geven en dan met haakjes openen een variabele mee te geven bijvoorbeeld cos(pi). Hier komt dan een waarde uit, namelijk de cosinus van pi. Hierin is cos een standaard functie.
* Je hebt ook al geleerd om je eigen functie te maken, door onder het tabblad editor te klikken en dan op function. Wanneer je dit doet krijg je bovenaan een balk met output, de naam van de functie en de inputs, zoals je hieronder kan zien.



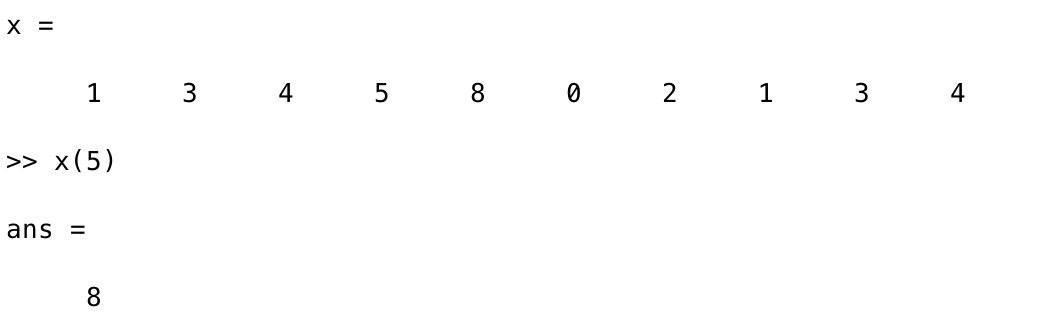
Figuur

* We gaan nu hieronder door functies met meerdere inputs en outputs. Succes!

# Functies met meerdere outputs

We beginnen met een functie die meerdere outputs geeft. Stel je wilt bij een 100 meter sprint van Usain Bolt weten op welk moment zijn snelheid maximaal is. Nu gaat er vast een lampje branden, vorige week heb je geleerd dat er een functie max(x) is. Waarmee je een maximale waarde kan berekenen. Als output krijg je dan een maximale waarde, maar we willen meer weten, namelijk bij welke afstand is dit? Dit kan je doen door te kijken naar een index nummer.

**Wat is een index?**  
Voordat we het gaan berekenen waar de snelheid van Usain Bolt maximaal is moeten we weten wat een index is. Index betekend letterlijk dat je waardes selecteert die horen bij bepaalde elementen. Je kunt het vijfde element van een vector x opvragen met het commando x(5). Het getal 5 tussen de haakjes is hier de index. Dus wanneer je van het onderstaande plaatje 5 stapjes telt zie je dat je bij 8 uitkomt.



Figuur Hierboven is een vector x te zien, doormiddel van de index op te vragen tussen haakjes krijg je de waarde 8 hieruit.

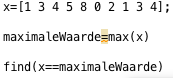
Je kan het ook op de ander kant op doen, stel je wilt weten welke index bij 8 hoort dan kan je het volgende doen (zie figuur 3). Je zoekt naar de waarde 8 en daar komt de index 5 uit.



Figuur

Het kan veel tijdkosten, wanneer je een hele lange vector hebt, om de maximale waarde te vinden. Door gebruikt te maken van de functie max() krijg je daar meteen de hoogstewaarde uit. Wanneer je die hebt kan je met find(x==’de gevonde waarde’) invullen en weet je het index cijfer van die maximale waarde.

**Voer het onderstaande commando uit, wat krijg je als antwoord?**



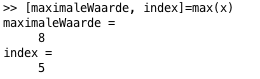
Figuur

Zoals je verwacht had zal je nu als indexwaarde 5 vinden. Maar dit kost veel programmeer regels en daarom is er bedacht om aan een functie meerdere outputs mee te geven. Dit kan je doen met blokhaken voor de functie aanvraag te zetten en ziet er als volgt uit:

[output1, output2] =functienaam(input)

Dit gaan we bekijken aan de hand van de max() functie. Bij de functie max() kan je naast dat je de maximale waarde eruit krijgt, ook meteen het indexnummer eruit krijgen van die maximale waarde.

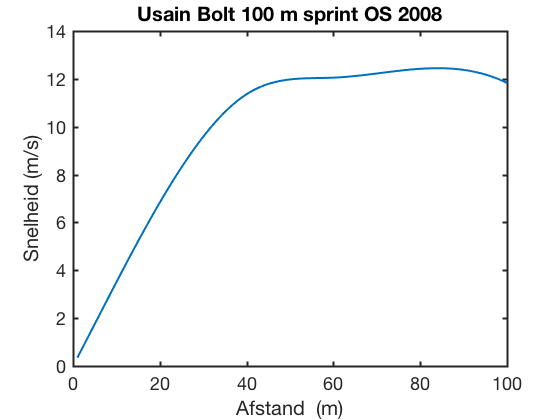
**Vul de onderstaande code in:**



Wat is de waarde die je eruit krijgt en wat is de index waarde?

Nu heb je een voorbeeld gezien met een functie waaruit meerder outputs komen, hoe kan je dit nu in de praktijk gebruiken?

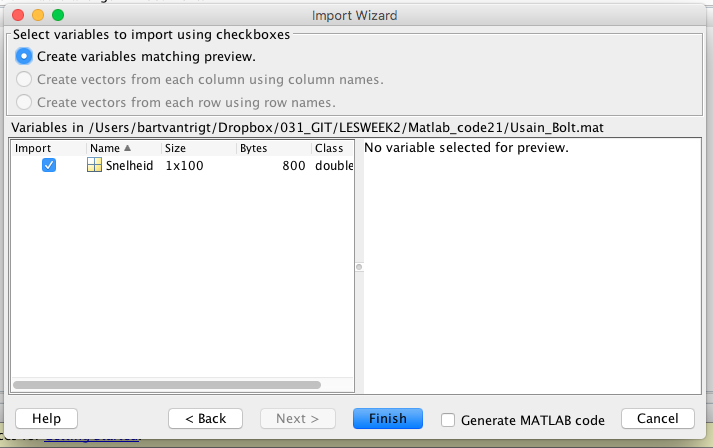
**Terug naar Usain Bolt.**  
Nu gaan we weer terug naar het voorbeeld van Usain Bolt. We hebben Usain Bolt zijn snelheid gemeten met versnellingssensoren, waar mee we door te differentiëren de snelheid krijgen (Dit ga je leren bij meten aan de mens in het volgende blok). In de figuur hieronder zie je de snelheid tegen de tijd van Usain Bolt op de 100 m tijdens de Olympische Spelen van 2012 in Beijing.



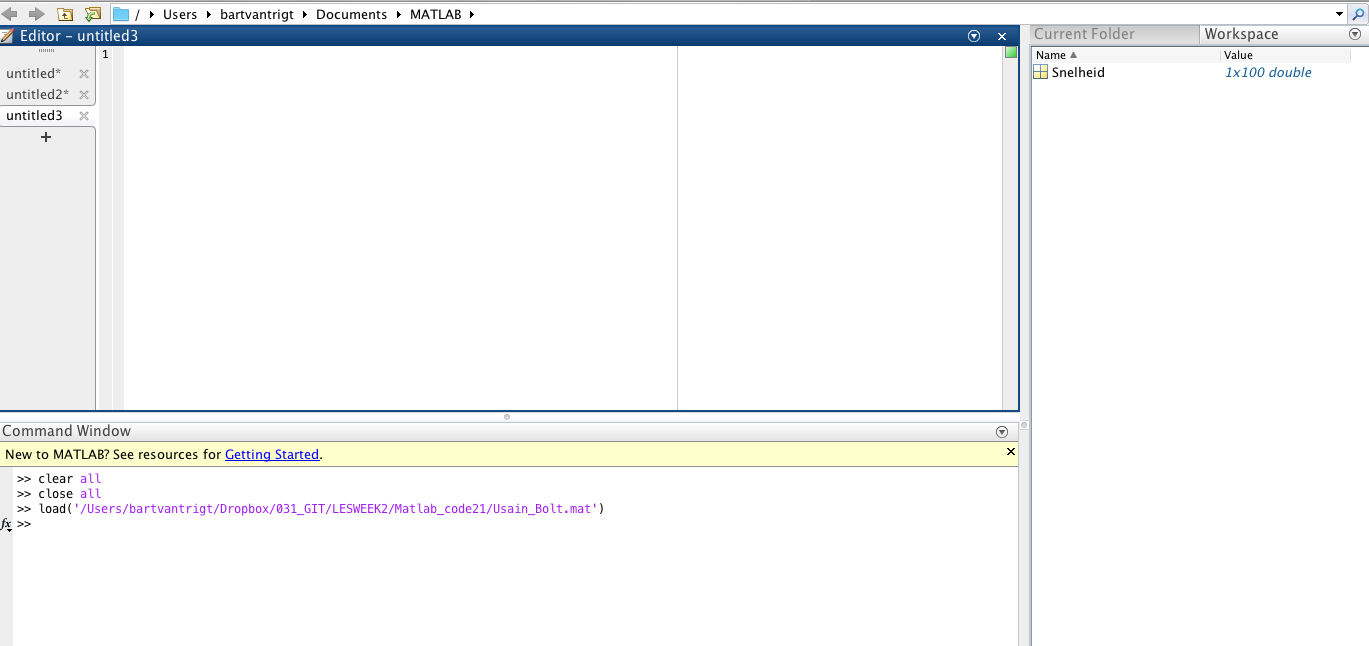
Figuur de snelheid tegen de afstand op een 100 meter sprint van Usain Bolt tijdens de OS van 2008 in Beijing.

**We willen nu bij de maximale snelheid weten hoe snel hij is. Dat gaan we als volgt doen:**

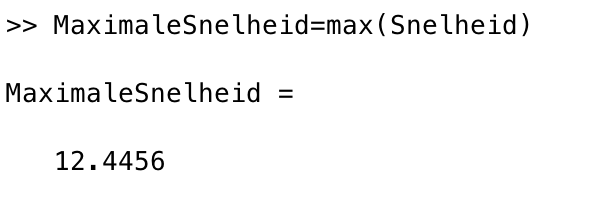
* Maak eerste je workspace leeg met clear all.
* De snelheid staat onder de naam Usain\_Bolt.mat op blackboard. Open het bestand Usain\_Bolt.mat door er dubbel op te klikken, vervolgens krijg je import wizard, zoals je hieronder ziet, klik dan op finish.



Nu verschijnt er een variabele met Snelheid in je Workspace, zoals je in de figuur rechtsboven kan zien



* Gebruik nu de functie max. Vul in maximaleSnelheid=max(Snelheid)



Figuur

We weten dus dat zijn maximale snelheid 12.44 m/s is, maar we weten nog steeds niet op welk moment. Dit kunnen we doen door nog een output te krijgen. Dit doe je door voor je functie aanvraag blokhaken te zetten, tussen de blokhaken kan je meerdere output variabele zetten, dat ziet er dan als volgt uit:

[maximaleSnelheid, index]=max(Snelheid)

Voer dit in in Matlab.  
Op hoeveel meter is Usain Bolt op zijn maximale snelheid? Klopt dit met het figuur 5. Laat je antwoord zien aan de docent

## Vragen en opdrachten

Beantwoord de volgende vragen met juist of onjuist:

1. We hebben y =[3 6 7 12 8 15 20 2 4]; Om het maximum te bepalen maken we gebruik van het commando max[y].
2. We hebben een commando max en willen het indexnummer weten dit doen we met het volgende statement: [hoogstewaarde,index]=max(x)
3. Je wilt meerdere outputs berekenen hiervoor gebruik je { output1, output2 }=min(x)
4. Voer het volgende commando in:

x = [20 11 11 5 10 23 14 8 8 20];

Bereken de index van het maximum en die van het minimum

1. Het kan voorkomen dat het maximum meerdere keren voorkomt.

Voer het volgende commando in:

x = [20 11 11 5 10 13 14 8 8 20];

## Antwoorden en uitwerkingen

1. Het antwoord is onjuist, het commando bevat ronde haken, geen blokhaken.
   1. Functies worden aangeroepen als volgt: max() whos () size(). Echter in deze vraag zie je dat de functie max wordt aangevraagd met behulp van blokhaken [], en niet van ronde haken. Deze stelling is dus fout omdat de functie een syntaxfout bevat.
2. Het antwoord is juist, met dit statement bereken je de hoogste waarde en de index.
3. Het antwoord is onjuist, je gebruikt blokhaken [ ] en geen accolade { }.

# Functie met meerdere inputs

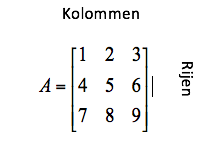
Net hebben we gezien dat een functie meerdere outputs kan hebben, je raad het al, een functie kan ook meerder inputs hebben.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Kolom 1 | Kolom2 | Kolom 3 |
| Rij 1 | 1 | 2 | 3 |
| Rij 2 | 4 | 5 | 6 |
| Rij 3 | 7 | 8 | 9 |

Bekijk de volgende matrix. Een Matrix kan gezien worden als soort tabel. LET OP! Dit is een versimpeling in de toekomst ga je meer leren over matrixen maar nu vatten we het even op als een tabel. Een matrix bestaat uit rijen en kolommen. In het vervolg van je studie ga je meer leren over matrixen.

Leuk weetje:

De eerste drie letters van MATLAB staat voor matrix en de laatste drie voor laboratory

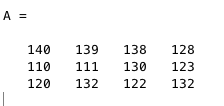


Het berekenen van het gemiddelde van een vector hebben we vorige week al gedaan dat doe je met het commando mean(x), de x is hier je vector. Van een matrix kunnen we de rijgemiddelden en de kolomgemiddelden uitrekenen. Echter, we zullen dit wel moeten aangeven. Een matrix/tabel heeft twee dimensies; de eerste zijn de rijen, de tweede de kolommen. Wanneer je de index opvraagt van bovenstaande matrix krijg je bij het volgende: Element A(3,2) is dan ook gelijk aan 8. Je gaat 3 rijen naar beneden en twee kolommen naar rechts. A(rijen,kolommen)

Stel je hebt een bloeddrukmeting gedaan bij drie proefpersonen, zoals je weet moet je een meting meerder keren uitvoeren wil het betrouwbaar zijn, dus je hebt vier keer hun systolische (boven)bloeddruk gemeten. De proefpersonen zijn de rijen en de metingen de kolommen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Meting 1 | Meting2 | Meting3 | Meting 4 |
| Mark | 140 | 139 | 138 | 128 |
| Bart | 110 | 111 | 130 | 123 |
| Hubert | 120 | 132 | 122 | 132 |

**Van de meetgegevens is de volgende Matrix gemaakt:**



Kopieer de onderstaande code naar je command window en je hebt dezelfde matrix.

A= [ 140 139 138 128; 110 111 130 123; 120 132 122 132]

Nu willen we het gemiddelde van de proefpersoon berekenen over de vier metingen. Dit kan je doen door twee inputs te geven aan de functie mean(input1,input2) dat wordt als volgt mean(A,2). Je geeft hierin nu de matrix A mee en met die twee geef je aan dat je de tweede dimensie wilt hebben dus dat er een kolomvector over moet blijven en deze is de tweede dimensie. Wil je de gemiddelde bloeddruk per proefpersoon weten dan middel je over een rij, dus de vier metingen.

**Vraag: Wie heeft de laagste bloeddruk en hoeveel is die?**  
antwoord: Bart met 118.5

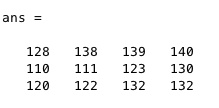
Je kan nu ook middelen over de kolommen, om te kijken of een meting ergens verkeerd ging. Zo blijft er een rijvector over. Matlab telt dan alle rijen bij elkaar op en middelt ze dan over de rijen. Dit doe je dan niet met mean(A,2) maar met mean(A,1). Dus over de eerste dimensie en dit zijn de rijen. Voer dit commando uit. Wat zie je?

Stel we willen de bloedrukmetingen mooi op volgorde zetten per proefpersoon. Hiervoor heeft Matlab ook een ingebouwde functie namelijk sort.

Voer het volgende commando in:



Je geeft nu zelfs drie inputs mee aan de functie. Namelijk je matrix, de dimensie en een string waarmee je zegt oplopend.



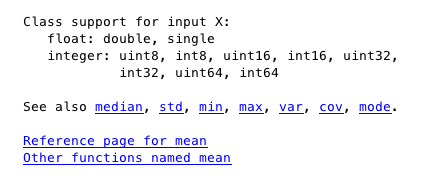
Je ziet nu dat de metingen mooi oplopen per proefpersoon. Probeer het ook eens met ‘descend’ wat zie je nu?

**De help functie!**

Vorige week heb je al gelezen over de help functie, maar omdat die zo belangrijk is herhalen we het hier nog een keer! Dit doen we omdat er zoveel functies zijn kunnen we nooit helemaal onthouden wat voor inputs en outputs er zijn. Daarom maken we gebruik van de help functie.

Voer maar eens in je command window: **help mean**

Er verschijnt nu een hele uitleg wat de input en outputs zijn van de functie. Om het wat overzichtelijker te maken kan je op reference page for mean klikken onderaan.

****

Dit kan je voor elke functie doen, de help functie is je grootste vriend, ook al is het soms lastig te lezen in het engels.

Weet je nog hoe je deze reference page ook kan openen uit je command window?

Probeer eens: **doc mean**

## Vragen en opdrachten

Beantwoorde de volgende vragen met juist of onjuist:

1. Een functie kan meerdere inputs hebben.
2. Wanneer je een index wilt oproepen van een Matrix gebruik je: A(kolommen,rijen)
3. Een functie kan verschillende inputs hebben, op dezelfde plek waarmee je hem aanroept.

Open vragen:

1. Wat is de relatie tussen de help functie en input?
2. Wat gebeurd er wanneer je maar 1 input invoert terwijl een functie wel meerder kan hebben?
3. Hieronder zie je een Matrix, hoeveel rijen en kolommen heeft deze matrix?



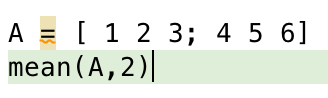
1. Bereken het gemiddelde van bovenstaande matrix per kolom.
2. Voer de a = [ 9 8 13; 4 3 15; 6 1 19] in, in Matlab. Hoe kan je het element 15 opvragen in Matlab? (Je mag niet spieken!)

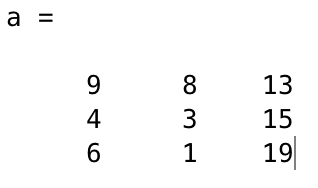
## Antwoorden en uitwerkingen

1. Juist
2. Onjuist
3. Juist

Open vragen:

1. Met de help functie kan je kijken wat de input is bij een specifieke functie.
2. Dan neemt de functie gewoon de default voor deze inputs.
3. Deze matrix heeft 2 rijen en 3 kolommen.



1. Als de code wordt ingevoerd geeft matlab de onderstaande matrix terug,

We zien dus dat 15 in de tweede rij staat en in de derde kolom. Weet je nog dat een matrix dimensies heeft? De eerste dimensie is voor rijen, de tweede dimensie zijn de kolommen.   
Het element kan dus worden opgevraagd aan de hand van a(2,3)

# Ingewikkeldere functies maken

Nu gaat het echt spannend worden (#FISSA), want we gaan een ingewikkeldere functie maken. We hebben net geleerd dat een functie namelijk meerdere inputs en meerdere outputs kan hebben.

Met Kinovea kan je de kniehoek berekenen van een fietser. We gaan nu een functie maken waaruit we meteen het gemiddelde, de mediaan, de maximale kniehoek en de minimale kniehoek kunnen bepalen van een fietser. De input data in je functie is de kniehoek in graden en je output data is je gemiddeldeKniehoek, mediaanKniehoek, maxKniehoek en minKniehoek in graden. Voordat je een functie gaat aanmaken is het handig om een schematisch beeld te maken van hoe je functie eruit komt te zien.

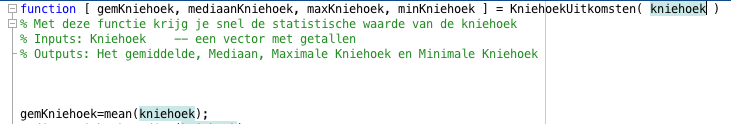
* 1. **Teken het onderstaande figuur na en vul in de figuur je input en je output variabelen in.**



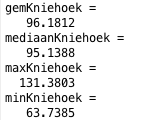
* 1. **Maak een nieuwe functie aan met de naam KniehoekUitkomsten.m en sla deze op.**

(Weet je niet meer hoe je een functie aanmaakt lees dit dan nog eens terug in de reader wk 1.2.)

* 1. De eerste regels zullen er ongeveer zo uitzien, de berekening van het gemiddelde is al gegeven



* 1. **Voeg nu de code toe om de mediaan, maximale kniehoek en minimale kniehoek te berekenen toe.**
  2. **Open het bestand fietsenkniehoek.mat of sleep deze naar je workspace. Hierin staat de variabele kniehoek en deze is gegeven in graden.**
  3. **Roep nu je functie aan in je command window, komen je antwoorden overeen met hieronder? Zo ja, dan heb je het goed gedaan, anders moet je je functie nog aanpassen**

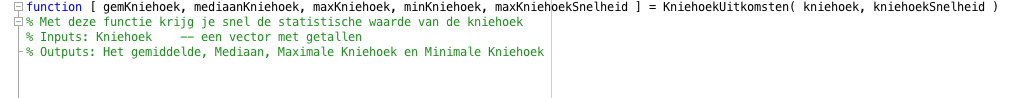
****

Stel we willen de maximale kniehoek snelheid weten. Daarvoor hebben we de kniehoek snelheid nodig en dus moeten we een extra input meegeven in de functie en ook een extra output.

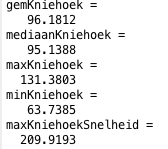
* 1. **Voeg deze toe aan je schematische figuur, zodat je het overzicht behoud.**



1. Je functie aanvraag zal er nu zo uitkomen te zien.



1. **Vul je functie verder met de code om de maximale kniehoek snelheid te berekenen.**
2. **Klopt je antwoord met hieronder wanneer je de functie opnieuw aanvraagt?**

****

## Vragen en opdrachten

Beantwoord de volgende vragen met juist of onjuist:

1. Wanneer je een functie maakt kan je zoveel inputs meegeven als je wilt.
2. Je kan een functie in een functie gebruiken.
3. Je kan een functie alleen aanvragen met je ‘command window’ en niet in een script

Open vragen:

1. Waarom gebruikt een BT’er standaardfuncties in Matlab?
2. Waarom maak je als BT’er je eigen functies?
3. Type in matlab de volgende code: VectorA = [1 3 15; 12 4 6; 3 5 7]. Roep nu de hulpfunctie voor max aan. Met behulp van welke code kan je het hoogste getal van VectorA oproepen, het getal 15?

## Antwoorden en uitwerkingen

Gesloten vragen:

1. Juist
2. Juist, dat heb je net gedaan bij het maken van je eigen functie.
3. Onjuist, je kan juist ook een functie aanvragen in je script, dit zorgt ervoor dat je netjes programmeert.

Open vragen:

1. Standaardfuncties besparen programmeurs en bewegingstechnologen veel tijd in het ontwikkelen van hun algoritmes. Dat is dus erg prettig.
2. Als BT’er komt het vaak voor dat je iets wilt berekenen, maar er geen standaard functie voor is. Daarom maak je je eigen functie om zo netjes te programmeren en dat je snel veel berekeningen kan uitvoeren in plaats van alles op papier uit te rekenen.
3. Als je de functie aanroept zie je dat hij veel verteld over het berekenen van het grootste getal, echter in Matlab staat niet uitgelegd wat de code is voor het grootste getal in de matrix. Je zal vast gedacht hebben: “ah ik moet vast de reference page for max gebruiken”. Onder het kopje ‘largest element in matrix’ is de code te vinden: *If you only need the maximum value of A and not its index then call the max function twice:* ***Het antwoord is max(max(VectorA)),***

# Specifieke terminologie en eigenschappen

We hebben nu veel geleerd over functies, in in dit hoofdstuk beschrijven een aantal Matlab termen en functies nader die het eenvoudiger maken om een functie te schrijven of data te analyseren. Probeer ze allemaal uit in Matlab zodat je precies ziet wat ze doen.

* **dimensie**Een vector of matrix heeft bepaalde dimensies, of afmetingen. In Matlab wordt met de eerste en tweede (en verdere) dimensies van een matrix specifiek de rijen (eerste dimensie) en kolommen (tweede dimensie) bedoeld.
* **end**   
  Hiermee geef je het laatste element van een vector aan. Voer het volgende prompt in in Matlab:

v= [ 3 5 6 8]

Dan is het resultaat van het commando v(end) gelijk aan 8. Dit is buitengewoon handig als je niet van tevoren weet hoe lang een vector zal zijn. Weet je nog wat je krijgt als je v(2) doet? Probeer het eens uit

* **Colon (:) (de dubbele punt)**

Om van een matrix een hele rij of een hele kolom op te vragen kun je een dubbele punt gebruiken.



A=[ 1 2 3 4; 5 6 7 8; 9 10 11 12]

* 1. Maak de matrix A aan in Matlab dit doe je door gebruik te maken van een puntkomma(;). Met een puntkomma ga je naar een nieuwe rij.
  2. Als we de derde kolom op willen vragen dan kunnen we dat doen met het commando A(1:3,3), rijen 1 t/m 3 van de derde kolom. Controleer je antwoord met hieronder:



* 1. Het volgende commando werkt ook, A(1:end,3), rijen 1 t/m de laatste van de derde kolom.
  2. Maar je kun dit nog eenvoudiger doen met het volgende commando: A(:,3). Dit geeft alle elementen uit kolom 3. Krijg je weer dezelfde antwoorden als hierboven? Dan gaat het goed! Probeer het nu eens voor de tweede kolom:
* **Index**

Deze term is uitgelegd in het hoofdstuk: ‘functie met meerdere inputs’

* **Transponeren**

Vooral bij vector- en matrixbewerkingen is het vaak nodig om de getransponeerde van een matrix te bepalen. Transponeren kan door een ’ achter een vector of een Matrix te zetten.

Wat krijg je als je dit doet voor de vector(v). Je ziet nu dat een rijvector door de ’ een kolomvector is geworden.

Misschien heb je weleens gehoord van ‘Meneer van Dale wacht op antwoord’. Deze regel gaat niet meer op.

5\*(3/6+0.5)

5\*3/6+0.5

**Komt hier hetzelfde antwoord uit?**

De bewerkingsvolgorde gaat als volgt:

* + 1. Haakjes
    2. Machtsverheffen en worteltrekken
    3. Vermenigvuldigen en delen
    4. Optellen en aftrekken

# Handige Matlab functies

Hieronder volgt een beschrijving van een aantal handige Matlab functies, er zijn er nog veel meer functies en deze zal je tijdens je opleiding allemaal nog gaan ontdekken, degene die hier besproken worden zal je het meeste gaan gebruiken.

Maak eerste een matrix(A) en een vector (v) aan in matlab, zoals je hieronder kan zien:

We gaan nu op deze data wat functies loslaten, voer voor elke functie de A en de v in. Erachter staat beschreven wat de functie doet.

* **size()**

Dit commando geeft de afmetingen van een vector of matrix. Voor de matrix geeft het commando size(A) de vector [2 4] terug. De matrix heeft dus 2 rijen en 4 kolommen. Voor de vector geeft size(v) de vector [1 4] terug; een rij en 4 kolommen. Je kan dit controleren door te kijken naar je workspace, komen ze overeen? Neem nu de transponent van A met een ’en bereken de size, wat is deze nu?

Antwoord: 3-bij-4

* **isempty()**

Dit commando geeft aan of een variabele leeg is. Als de variabele leeg is, krijg je de logische waarde 1 terug. Als de variabele *niet* leeg is, krijg je een logische nul terug.

* **numel()**

Dit commando geeft het aantal elementen van een variabele aan. Hoeveel is dat voor de matrix A en hoeveel voor die van v? Weet je het antwoord niet uit je hoofd? Geen probleem voer het in in Matlab!

* **find()**

Find is een hele handige functie, omdat je hiermee de indices kan achterhalen van elementen met een specifieke inhoud. **Vul in** find(v==8) geeft alle indices van de vector v waar een 8 staat. Stel je wilt alle waardes groter dan 6 weten wat krijg je wanneer je find(v>6) invult? **Controleer je antwoord door het in te vullen in matlab.**

* **strcmp() en strcmpi()**

Met deze commando’s kun je bepalen of twee strings aan elkaar gelijk zijn. Maak twee variable aan in de vorm van een string met je naam met een hoofdletter en de andere met een kleine letter.

A= ‘Bart’

B= ‘bart’

Run nu de volgende twee codes.

strcmp(A,B) en strcmpi(A,B)

**Wat is het verschil dat je eruit krijgt? En wat betekent het? Weet je het niet zoek het op in de help functie.**

Het commando strcmp() houdt wel rekening met hoofdletters, het commando strcmpi() is ongevoelig (*insensitive*) voor hoofdletters.

## Vragen en opdrachten

1. Maak in Matlab een nieuwe variabele aan S1 = ‘kiwifruit1234’ en een S2 = ‘kiwibird123’. Roep de strcmp() functie aan zodanig dat je de string S1 vergelijkt met S2. Wat krijg je als antwoord terug? Wat betekent het getal dat Matlab teruggeeft?
2. Size() geeft de afmetingen van een vector of matrix. Stel de functie geeft het volgende terug ans = 2 3. Stelling: dit betekend dat de matrix 2 kolommen en 3 rijen heeft. Waar/ niet waar.
3. Maak een string aan met je naam, bijvoorbeeld str = ‘Bart’; Bepaal de afmetingen en het aantal elementen.
4. Bekijk de help tekst van het commando find en bepaal op deze manier waar in de vector

v = randi (20,1,20); de getallen met waarde groter of gelijk aan 10 zitten.

1. Wat is het verschil tussen de volgende twee commando’s:
   1. find(v==10);
   2. v==10

## Antwoorden en uitwerkingen

1. De variabele S1 heeft 13 karakter, en S2 heeft er 11. Strcmp() geeft terug of de strings evenveel karakters hebben, wat in dit geval dus niet zo is. Strcmp() geeft een 0 terug, oftewel een 0 is false, is de variabele zijn niet gelijk aan elkaar in karakters
2. Dit is onjuist. In H5 wordt uitgelegd dat de eerste dimensie de rijen zijn en de tweede dimensie de kolommen. Ans = 2 3 betekend dat de matrix 2 rijen heeft en 3 kolommen.
3. Als het goed is, krijg je voor elke letter een cijfer, dus voor Bart krijg je 1 bij 4 eruit
4. Dit kan je doen met het statement find(v>=10)
5. Wat zie je als je het in Matlab uitvoert? Dat is dus het verschil.