Programmeertaal met krachtig statistisch arsenaal

Analyseren en voorspellen met R

Rick van Rein

De taal 'R' duikt steeds vaker op in de literatuur rond analyse van gegevens, zoals die in een database. Voor het ontdekken van trends en voorspellen van toekomstige ontwikkelingen is het dan ook een interessante tool. Maar wat is dat nu, R?

Veel van wat met R kan worden gedaan gaat over hoe handig de taal zich ergens voor leent, meer dan dat het nu zulke unieke eigenschappen heeft die het een heel eigen taal maakt. In R schrijf je dingen snel en eenvoudig op, en het werkt handig om wiskundige analyses mee te prototypen, bijvoorbeeld om vermoedens te testen of verschillende parameters in een trend uit te proberen.

Gewoon een programmeertaal

Als je het heel strikt bekijkt, is R eigenlijk vooral een programmeertaaltje. Het is de open source herimplementatie van S, dat in 1976 door Bell Labs is ontwikkeld. Deze taal was vooral bedoeld voor numerieke manipulaties en het maken van plotjes van functies zoals tijdreeksen of het meer algemene verband tussen variabelen.
Het soort werk dat je snel met R kunt doen lijkt wel iets op wat je ook in een spreadsheet kunt doen, maar de interactie is minder op een muis gebaseerd en gaat wat meer richting programmeren. Grafisch met een muis werken is best handig als je iets voor het eerst doet, maar als je vaak bepaalde dingen moet doen gaat er niets boven een tool die je niet altijd als een beginner behandelt, maar je de kans biedt te groeien.

Zaken die uitstekend zijn uitgewerkt in R zijn statistische modellering, zoals lineaire regressie volgens een veelvoud van modellen, en het werken met array's en matrices. Ideaal als je een wat wiskundige inslag hebt dus. Dat sluit het nut van R overigens niet uit voor wat simpeler rekenwerk, zoals het doorrekenen van een hypotheek.
In mijn opleiding heb ik ooit leren werken met APL, ook zo'n taaltje dat veel met matrixbewerkingen deed, met als toepassingsgebied de modellering van digitale schakelingen zoals processoren. Dat werkte aardig, maar wel met een heel eigen instructieset die bestond uit symbolen die de rest van de wereld niet gebruikte. In dergelijke toepassingen is R's gebruik van standaardsymbolen een verademing, en inderdaad zou deze simulatie van hardware even goed met R kunnen worden gedaan.

Reeksen en matrices

Iets wat supersimpel is in R, is het rekenen met meerdere waarden tegelijk. Zoals een functie een algemeen voorschrift f(x) is voor willekeurige inputwaarden x, zo is het met R mogelijk om een berekening toe te passen op een array van waarden op dezelfde manier als op een enkele waarde. Als in x een enkele waarde staat dan rekent f(x) daarvoor de uitvoer, maar als x een array is dan wordt de berekening toegepast op alle variabelen in x. Bijvoorbeeld:

```
x <- 0:360 * pi/180
y <- 230 * sqrt (2) * sin (x)
z <- 230 * sqrt (2) * cos (x)
```

De extreem korte notatie 0:360 voor de reeks getallen 0, 1, ..., 360 is erg handig, net als de mogelijkheid om er onmiddellijk mee door te rekenen. In dit voorbeeld worden alle gehele graden van 0 tot en met 360 omgezet in een hoek in radialen, waarna de waarde y wordt uitgerekend. De uitkomst is een sinusgolf die zich ook nog eens heel simpel laat tekenen:

```
plot ( y ~ x )
```

Hierin is ~ te lezen als 'tegen', dus dit is een plot van 'y tegen x' ofwel met horizontale waarden uit x en verticale uit y. Het is mogelijk hier nog dingen bij in te tekenen, zoals

```
points ( z ~ x )
```

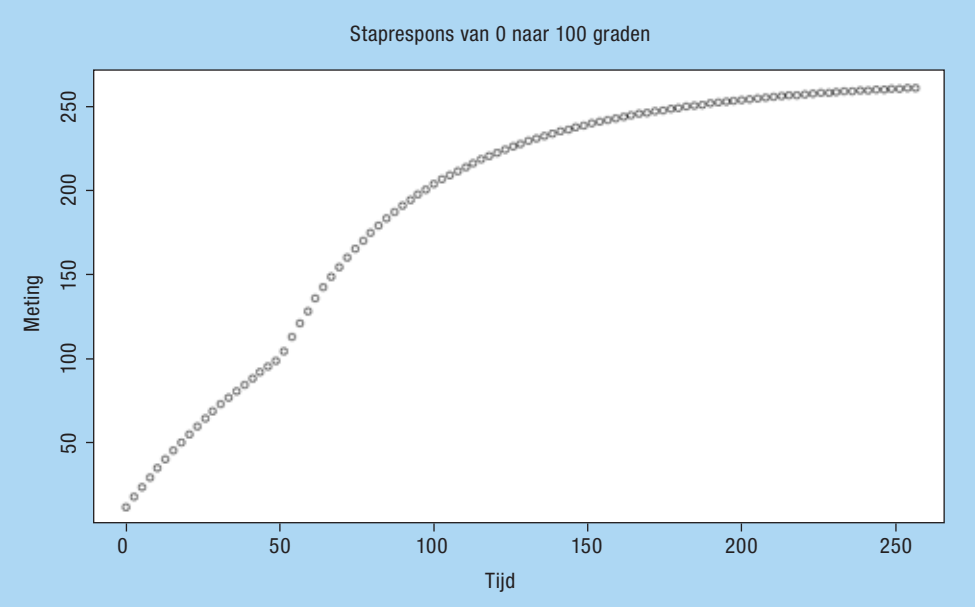
Natuurlijk is het ook mogelijk om andere variabelen tegen elkaar uit te zetten. De volgende plot levert bijvoorbeeld een cirkel:

```
plot (z ~ y)
```

Al deze plots werken prettig op de commandline van R, dus interactief. Maar uiteraard kan er ook gescript worden, en kunnen de plaatjes worden gedumpt in bestanden in een keur aan formaten. Dat is handig wanneer een trendanalyse is uitgekristalliseerd en dagelijks uitgevoerd moet worden.

Voorbeeld: staprespons

Een aardig voorbeeld van wat R kan is het herkennen van stapresponses. Dat is de meetbare uitwerking van een scherpe



Afbeelding 1: De vertraagde respons op een stapvormige verwarming van de leidingtemperatuur.

stap ergens binnenin een systeem, maar die door bufferwerking en vertragingen veel zachter naar buiten treedt. Een complexe toepassing zou kunnen zijn hoe snel geld wordt opgenomen als een bank failliet gaat. (Absurd is overigens dat dit met DSB gebeurde een dag nadat ik dit voorbeeld bedacht.) Maar wegens de beperkte ruimte gebruiken we hier een wat bescheidener voorbeeld.

Neem bijvoorbeeld een zonnecollector die opeens in zonlicht baadt, maar waarvan de gemeten temperatuur van de leidingen langzamer stijgt doordat de leidingen tussen het water en de meter eerst moeten opwarmen. Door de staprespons omgekeerd toe te passen kan worden teruggerekend hoe snel het water opwarmt vanuit een gemeten vertraagde temperatuur. In afbeelding 1 is te zien hoe de staprespons is voor een leiding die van 0 naar 100 graden 'stapt'.

De analyse vervolgt zich nu door uit de staprespons een impulsrespons af te leiden (afbeelding 2), dat wil zeggen wat de reactie is op 1 meetperiode lang 1 graad hoger in de leidingen, en dit kan daarna omgekeerd worden toegepast om uit de meetgegevens de interne leidingtemperatuur af te leiden. Dit alles gaat eenvoudig in R, en doordat het programmeerregels zijn (en geen grafische handelingen) is het eenvoudig om de handelingen te loggen voor later hergebruik. Bovendien is het eenvoudig om van allerlei tussenresultaten snel even een grafiekje te tonen – debuggen is dus juist wel grafisch, en dat maakt dergelijk analysewerk bijzonder interactief en dus snel.

Het uiteindelijke proces is te noteren als vermenigvuldiging van de laatste metingen met een in de tijd omgekeerde impulsrespons. Dat soort constructies staat in de meet- en regeltechniek bekend als een DSP-systeem, want eigenlijk zijn we met dit proces dat type meetcorrectie aan het prototypen.

De resultaten liegen er niet om. In afbeelding 3 staat zowel de actuele meetwaarde als de daaruit afgeleide interne temperatuur die daarvan de oorzaak is. We kijken als het ware door de vertragende effecten van de leidingen en het meetsysteem heen!

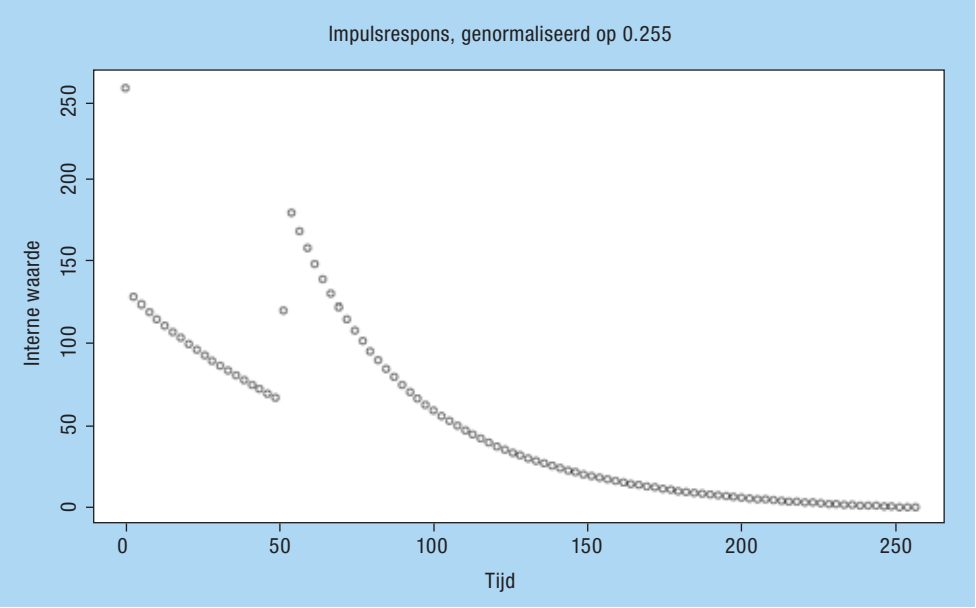
Dit voorbeeld geeft een fraaie toepassing om bestaande trends volgens een bepaalde theorie te voorspellen, en te ontdoen van ruiselementen. En dat alles is voor een bedreven gebruiker met R zeer snel te doen.

In de praktijk, die vaker financieel en complex is dan technisch, is het even goed mogelijk om te analyseren. Zo kan bijvoorbeeld een model worden geprobeerd, om te zien hoe goed het 'matcht' met iets als omzetontwikkeling over de tijd. De statistische functies van R helpen dan om te bepalen waar de grootste afwijkingen zitten, of welk model het beste functioneert, enzovoort. Een model extrapoleren naar de toekomst is vanzelfsprekend ook mogelijk, inclusief (voor wie stevig in de statistiek verankerd zit) een inschatting van de onzekerheid van de voorspellingen.

Een model extrapoleren naar de toekomst is vanzelfsprekend ook mogelijk

Het nut van matrices in dergelijke modellen is groot omdat het kan helpen de afhankelijkheid van meerdere variabelen in beschouwing te nemen. Ook hiervan is de taal goed doortrokken, zodat het ook in de praktijk snel en simpel werkt.

Het is geen wonder dus dat R steeds vaker opduikt als het gaat om het ontwarren van trends uit een kluwen gegevens. En dan is in het bovenstaande nog niet eens gebruikgemaakt van de weelde aan statistische standaardbewerkingen die R rijk is! Lineaire regressie, spline-interpolatie, generieke optimalisaties. Daarnaast



Afbeelding 2: De impulsrespons die overeenkomt met de staprespons van afbeelding 1.

voorziet deze functionaliteit ons natúúrlijk van statistische verantwoording door middel van standaardafwijkingen en *residuals*, zekerheidsintervallen en *p-values* – want het is wel leuk om vooruit te kijken, maar daarbij wil je wel graag weten hoe mistig het uitzicht is. Om te voorkomen dat je handelt met meer vaart dan de remweg billijkt.

Modellen, tabellen, kolommen

Een centrale constructie in R is het model. Het is bijvoorbeeld goed mogelijk hierin databasetabellen onder te brengen. Elke kolom in de database wordt dan in een afzonderlijke modelvariabele bereikbaar gemaakt. In wezen is een model wat in andere talen een *record* of *struct* heet: een verzameling van bij elkaar horende waarden met elk hun eigen type en een label om ernaar te verwijzen. In een model m kunnen bijvoorbeeld variabelen m$x en m$y zitten. In de toepassing van R om een database via een model te benaderen zouden x en y elk een databasekolom zijn. Dit geeft R een kolombenadering van de data, iets wat vaker voorkomt in de data-analyse. Doordat het in R mogelijk is om variabelen met zo'n kolom in een berekening te vermelden alsof het normale losse waarden betreft, zijn bewerkingen binnen een record (zoals die ook achter een SELECT zouden kunnen staan) gewoon op te schrijven in R als waren het elementaire berekeningen – R vouwt het dan vanzelf uit over al die elementen in de afzonderlijke kolommen.

Als een kolom te kort is dan wordt de inhoud daarvan net zo lang herhaald tot alle kolommen tenminste eenmaal helemaal afgewerkt zijn. Dit zal met de toepassing als kolomdatabase niet snel voorkomen, tenzij er losse getallen voorkomen; dit zou het geval kunnen zijn wanneer bijvoorbeeld totalen of maximalen uit een databasekolom worden bepaald. Of uit een berekening op zo'n kolom.

Verwant aan de update-statements is er ook een interessante formulering voor records waaraan iets zou moeten veranderen, tussen vierkante haken als ware het een index:

```
m$x [m$y > 0] <- 13
```

Wat hier staat is dat m$x en m$y beide stuk voor stuk worden afgelopen. Voor elk wordt gekeken of m$y > 0 is en zo ja dan wordt dat element van m$x overschreven met m$x + m$y. In wezen komt dat op hetzelfde neer als de SQL-formulering

```
UPDATE m
SET m.x = 13
WHERE m.y > 0
```

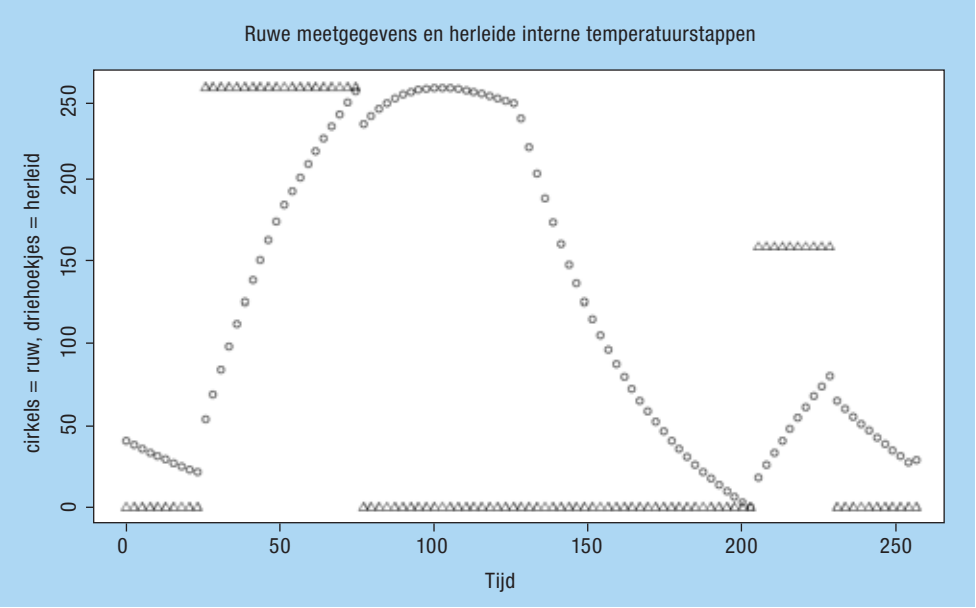
Veel constructies van SQL hebben zo een tegenhanger in R. Dat is ook niet zo verwonderlijk, want in beide modellen worden variabelen binnen de context van een record naast elkaar gezet om bewerkingen mee te doen. Hoewel niet onmiddellijk handiger dan de tegenhanger in SQL, en zeker voor meer mensen verder van huis, is dit in elk geval faciliterend voor het gebruik voor data-analyse.

Imperatief programmeren

De stijl van programmeren in R is imperatief. Assignments van uitkomsten van berekeningen, de nodige taalconstructies voor flow control, en expliciete opdrachten om te plotten zijn daarvan de oorzaak. Voor interactief werken is dit een model dat intuïtief werkt. Het is niet het enige model, het is niet zaligmakend, maar het werkt en is snel duidelijk.

Handen en voeten

Mocht R zo langzamerhand zijn gaan klinken als een algemene programmeertaal die uitblinkt in bepaalde zaken, dan moeten we hier toch wat gas terugnemen. Hoewel er veel is wat met R



Afbeelding 3: Hier staan cirkels voor de ruwe meetgegevens, terwijl de driehoekjes aangeven welke interne temperatuur daarbij hoort. Een simpele correctie in R op basis van wat wiskundige achtergrond maakt dit soort analyses snel mogelijk, en ook snel te ontwikkelen. Alle Y-schalen zijn overigens op een interval 0:255 genormaliseerd.

kan, haalt R het niet in vergelijking met talen als Python en Perl als het aankomt op integratie met een omringend systeem. Hoewel de taal op een aantal manieren kan worden aangeroepen die scripten in de hand werkt, inclusief zelfs binnen een webomgeving, is dat niet heel vanzelfsprekend. De opstart van R, die tot enkele seconden duurt, is al een eerste struikelblok om er alledaagse scripts mee te maken. Voor een uitgebreide analyse is dat al veel minder een probleem, maar het beperkt de range van toepassingen van R tot interactieve en batchgewijze bewerkingen. Wat dat betreft is R dus minder geschikt als taal voor algemene toepassingen.
Ook het verwerken van grote hoeveelheden data is niet erg praktisch, daar is R gewoon niet op ontworpen. Heel praktisch sluit dit datamining al uit als toepassingsgebied. Het uitstippelen van trends is een ander verhaal, doordat hierbij soms veel rekenwerk moet worden gedaan aan relatief weinig data. Met name wanneer interactief vermoedens worden getoetst dan is dit handig. De statistische kracht kan bovendien heel nuttig zijn om te onderzoeken hoe goed een vermoede trend ook werkelijk klopt met historische gegevens.

Er is rond R een expliciete keuze gemaakt om de invoer en uitvoer van gegevens niet enorm te ontwikkelen, zoals dat wel het geval is bij de populaire scripttalen. De redenatie is dat de omgeving waarin R draait vaak voldoende faciliteiten heeft om data om te zetten in een vorm die voor R acceptabel is. Dat is waar, maar hiermee reduceert R zich wederom tot batchprocessen, en wordt het lastiger om live in te spelen op real-time processen. Wie een website bouwt waar men boeken kan bestellen zal R niet snel te hulp roepen om de smaak van de bezoeker in te schatten en een paar extra suggesties te doen. Op zich is dat jammer, want het instrumentarium is verder ronduit indrukwekkend.

Al met al wordt R hierdoor gereduceerd tot een taal voor batchprocessen die redelijk veel rekenwerk vereisen maar niet op extreem veel data. Het is overigens mogelijk om de werkwijze van R mee te nemen naar andere talen. Zo is er voor Python een pakket dat R aanspreekbaar maakt vanuit die scripttaal, met een keurige vertaling van alle gebruikte datastructuren. In feite is dat slechts een *wrapper* rond R, en doet het niet af aan de beperkingen die bij R horen.

Conclusie

R is een taal die zich goed leent voor een bepaald toepassingsgebied. In dat gebied vinden we trendanalyse van niet al te grote datasets. Waarin R met name uitblinkt is interactieve toetsing van theorietjes en vermoedens. Het krachtige statistische arsenaal en de manier van werken die snelle plotjes oplevert zorgen daarvoor. Ook is zoiets om te zetten in een batchproces dat met enige regelmaat nieuwe grafieken oplevert.

Verder dan dit lijkt R niet te kunnen of willen kijken. Het is niet voor grote datasets geoptimaliseerd en is daarmee ongeschikt voor datawarehousing. Het integreert niet optimaal met de omgeving waarin het draait, en mist daardoor soms kansen. Tenslotte is er een vertraging van enige seconden tijdens de opstart van R, wat het ongeschikt maakt voor allerhande interactieve processen.

Noot

www.r-project.org is de uitvalsbasis van R, inclusief een rijke verzameling modules die de taal versterken met extra functionaliteit.

Rick van Rein

Dr. Ir. H. van Rein (rick@openfortress.nl) is ontwikkelaar en beheerder bij OpenFortress Digital signatures.