Wiskundige modellering in de ingenieurswetenschappen: Bordoefeningenles 4

▶ Oefening 1

▶ Oefening 2

▶ Oefening 3

▼ Oefening 4

Noisy Freefall

```
> restart:with(LinearAlgebra):with(plots):with(MTM):
```

Rijen en kolommen (# datapunten per meting en # metingen)

```
> mmax := 100:
  nmax := 200:
```

Functies: constante functie, lineaire en kwadratische functie

```
> f1 := t -> t^0:
  f2 := t -> t^1:
  f3 := t -> t^2:
```

Tijdpunten (equidistant array/sequence)

```
> ts := [seq(t, t=0..mmax, 1)]:
```

Random generators: R1, R2 voor random beginsposities, beginsnelheden
                    H   voor random noise per meetpunt

```
R1f  := rand(-100000.0..100000.0):
R2f  := rand(-1000.0..1000.0):
H    := rand(-1600.0..1600.0):
```

lijst van beginposities en beginsnelheden

```
> R1 := [seq(R1f(), i=0..nmax)]:
  R2 := [seq(R2f(), i=0..nmax)]:
```

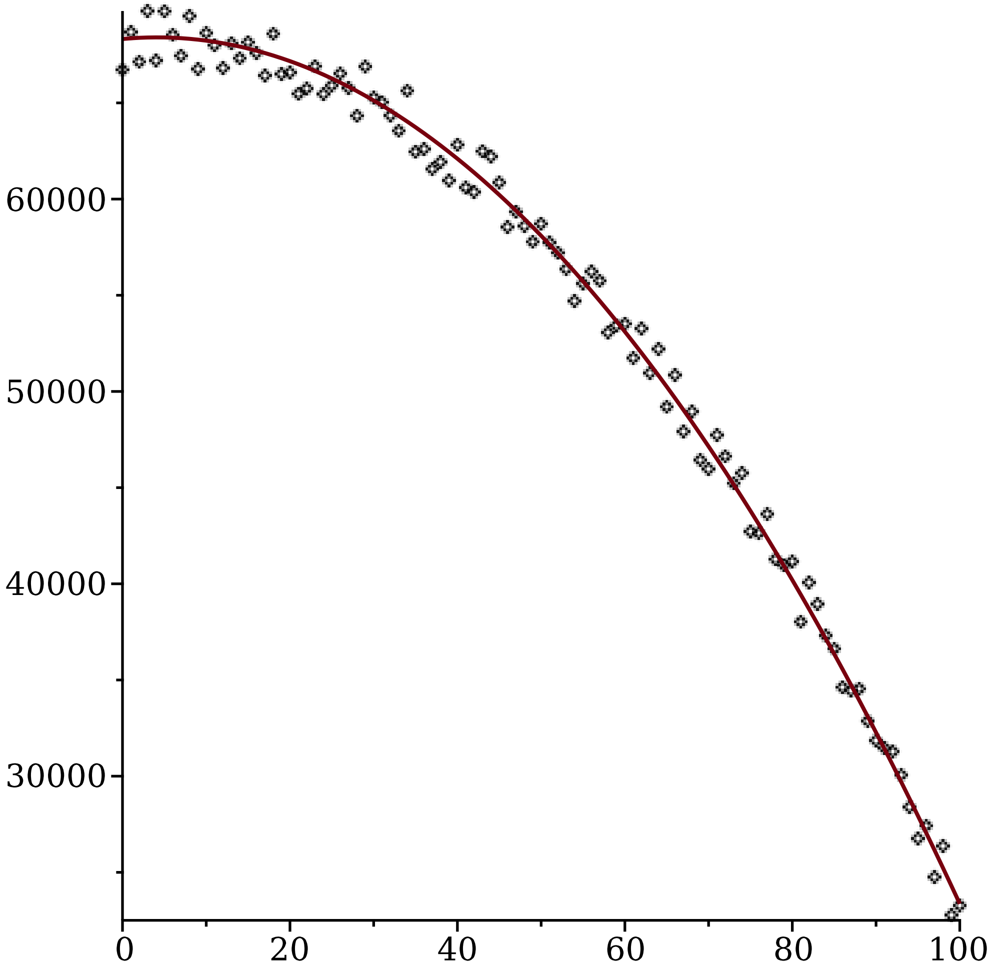
Datamatrix (metingen in kolommen van M)

```
> M  := Matrix(mmax+1, nmax, (i, j) -> R1[j]*f1(ts[i]) + R2[j]*
  f2(ts[i])-9.8/2*f3(ts[i])+H()):
```

visualisatie meting i

```
> i := 101;
  pointplot_i := pointplot(ts, M[..,i]):
  curveplot_i := plot(ts, R1[i]*map(f1,ts)+R2[i]*map(f2,ts)
  -9.8/2*map(f3,ts)):
  display(pointplot_i, curveplot_i);
```

$$i := 101$$



SVDecomposition

```
> U,S,V := svd(M):
```

Singular values: 3 grote (orde 10^6, 10^5), rest zijn kleiner (orde 20)

```
> Diagonal(S);
```

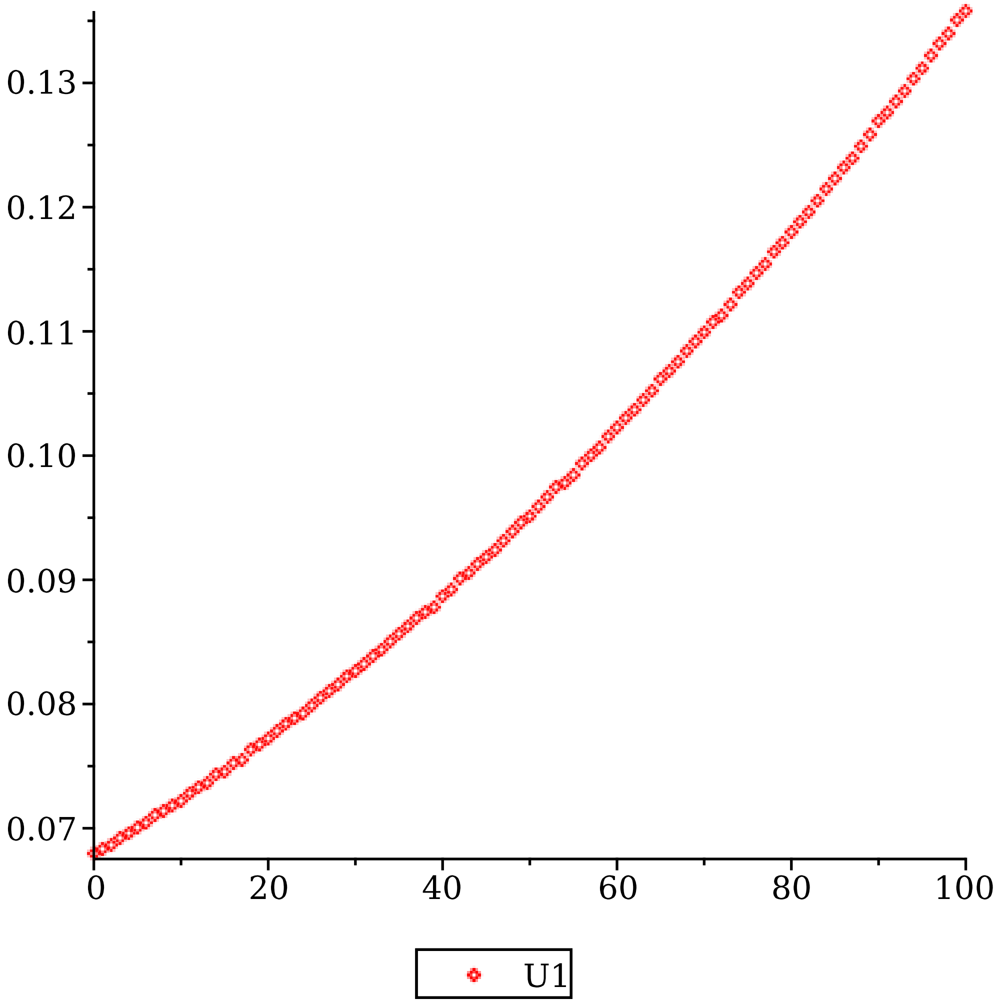
(4.1)

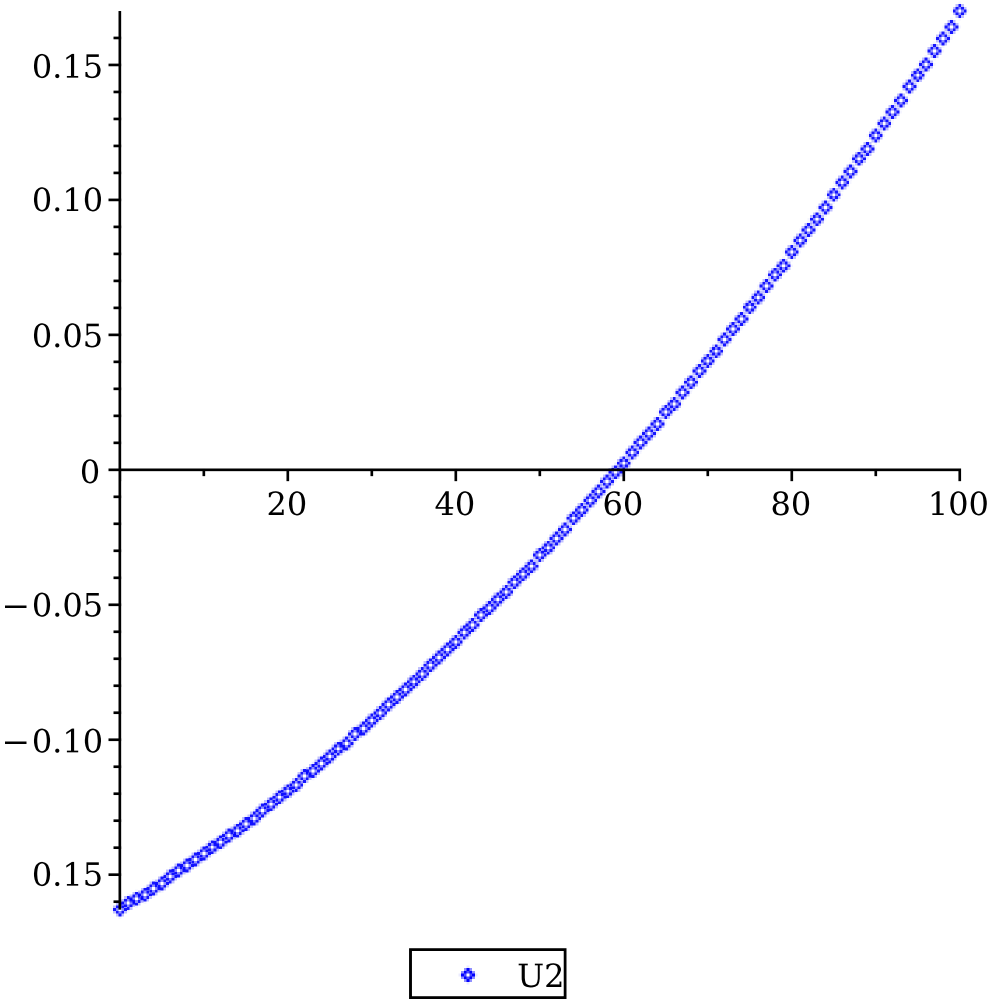
$$\begin{bmatrix} 9.64056374121872 \times 10^{6} \\ 2.73723132122265 \times 10^{6} \\ 396624.491513215 \\ 22195.3215958040 \\ 21414.8781379108 \\ 20743.5564482751 \\ 20522.3370051525 \\ 20028.2625394204 \\ 19723.4939016135 \\ 19679.6542984042 \\ \vdots \end{bmatrix} \tag{4.1}$$

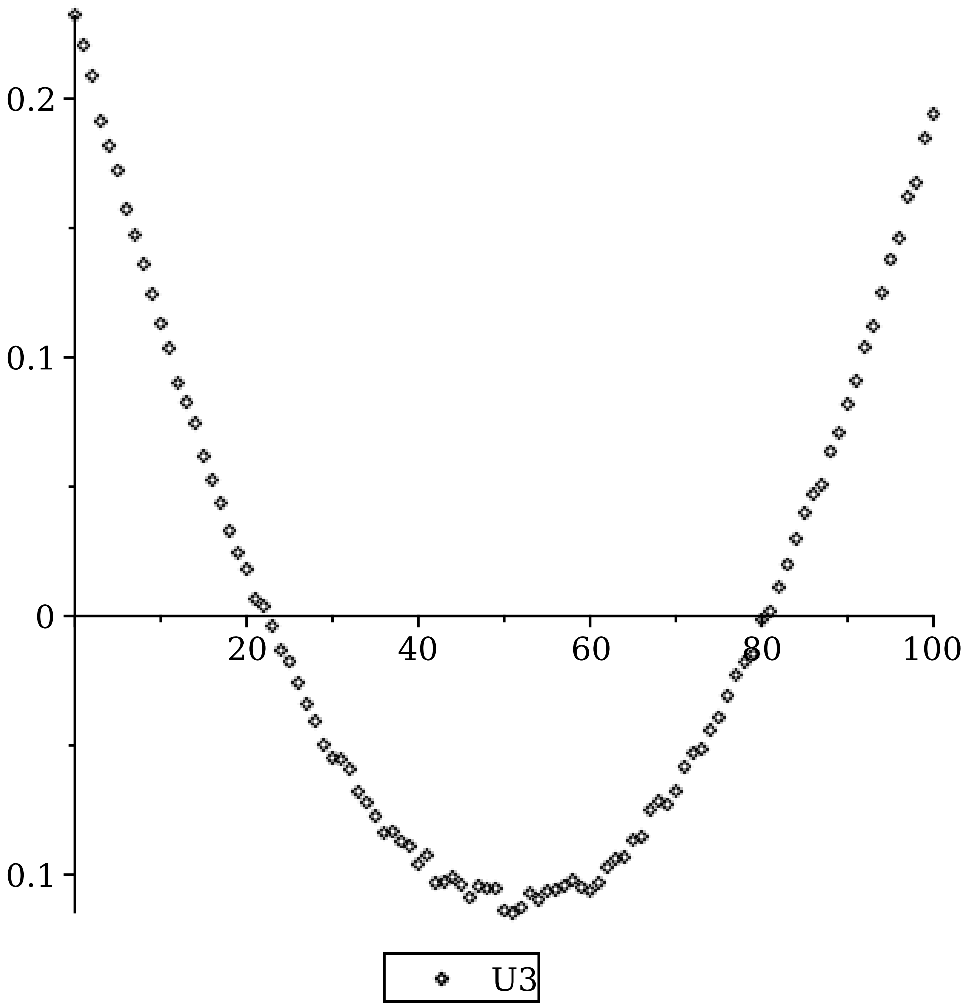
101 element Vector[column]

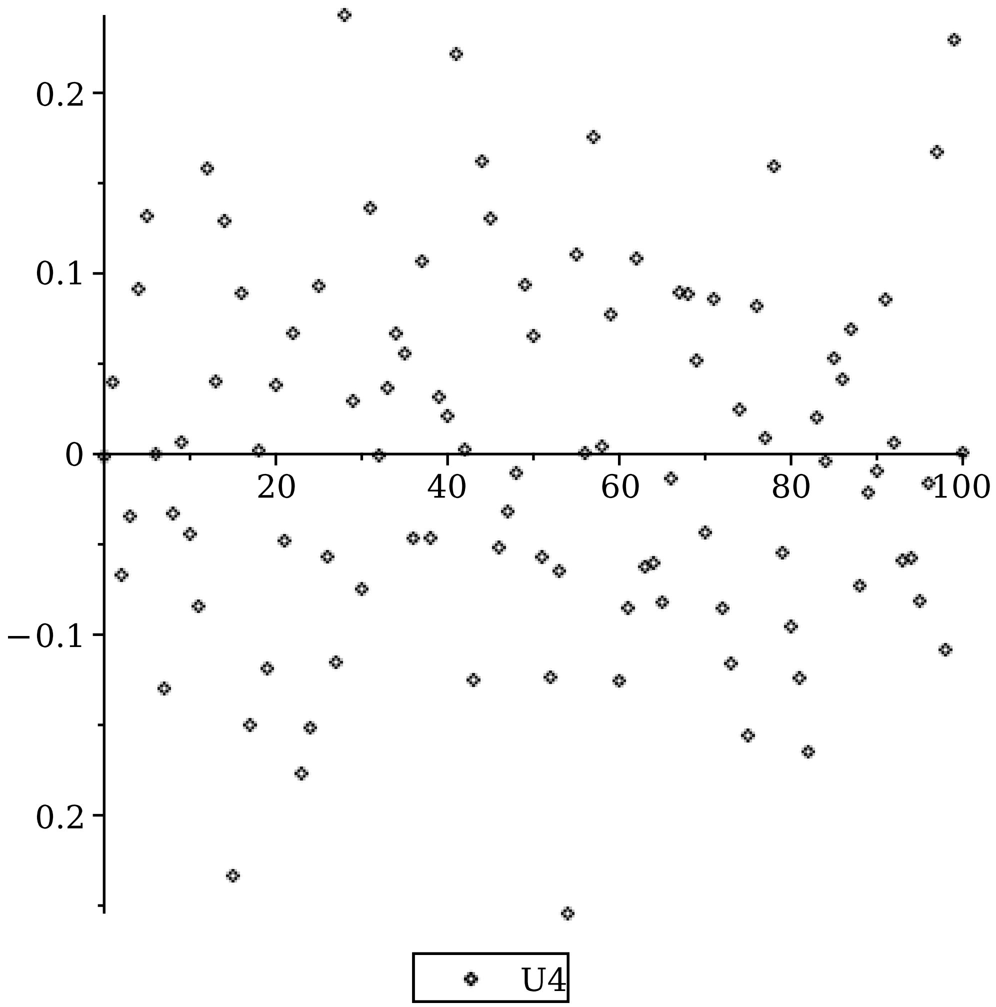
Plot U vectoren bij eerste 3 singular values

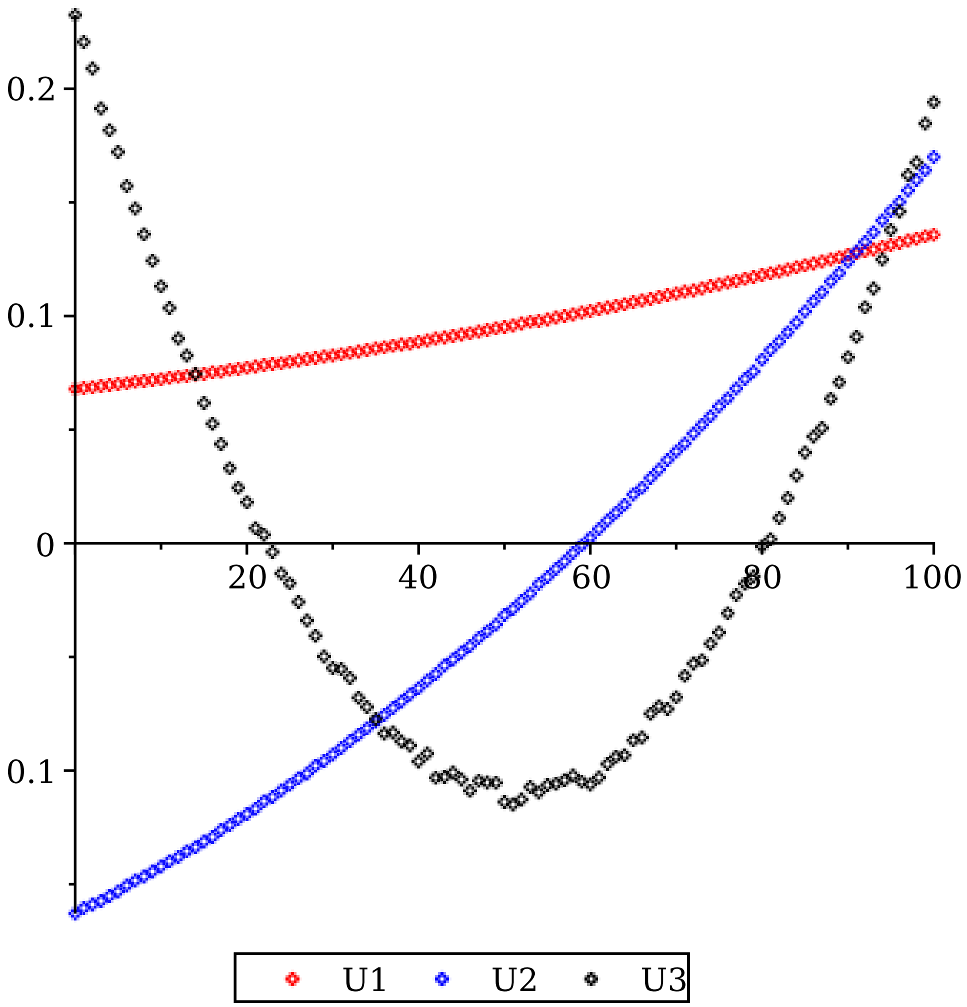
```
> U1plot := pointplot(ts, U[..,1], color=red,   legend="U1");
  U2plot := pointplot(ts, U[..,2], color=blue,  legend="U2");
  U3plot := pointplot(ts, U[..,3], color=black, legend="U3");
  U4plot := pointplot(ts, U[..,4], color=black, legend="U4");
  display(U1plot,U2plot, U3plot);
```


0.13
0.12
0.11
0.10
0.09
0.08
0.07
0
20
40
60
80
100
U1


0.15
0.10
0.05
0
−0.05
−0.10
0.15
20
40
60
80
100
U2

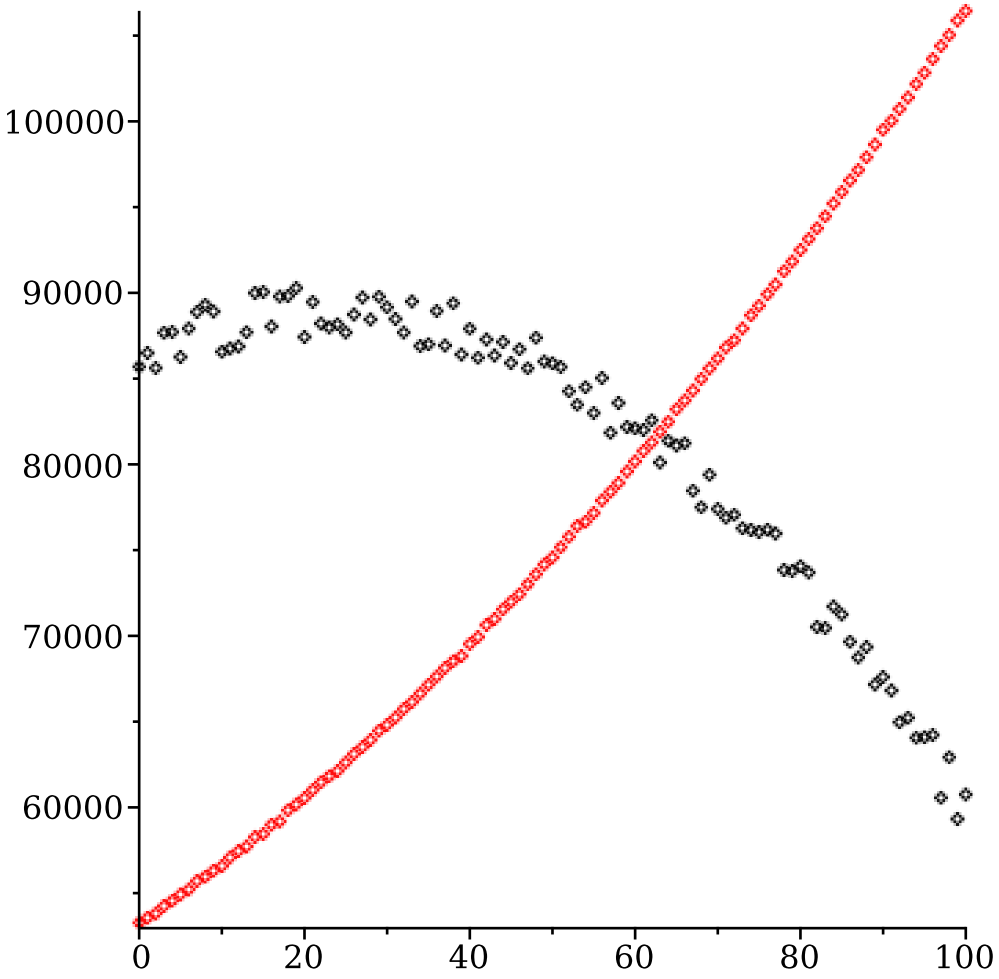

0.2
0.1
0
0.1
20
40
60
80
100
U3

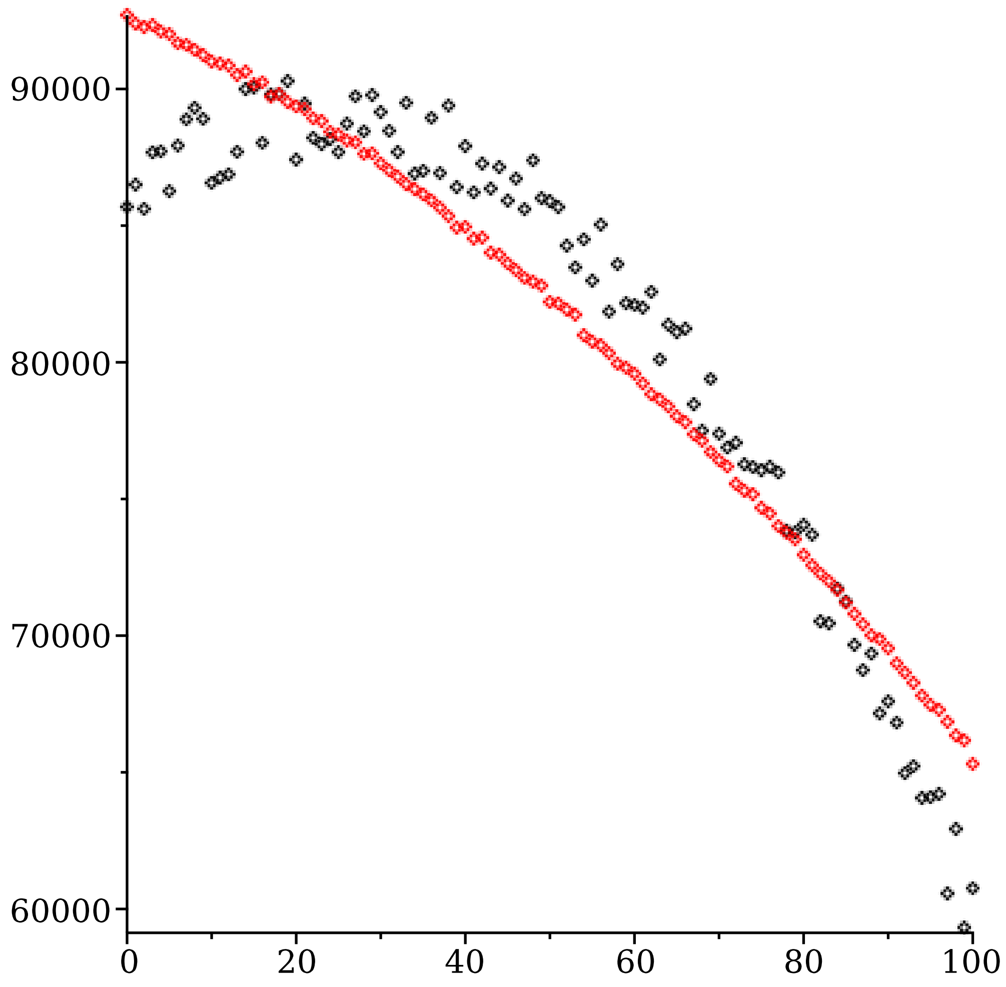

0.2
0.1
0
−0.1
0.2
20
40
60
80
100
U4

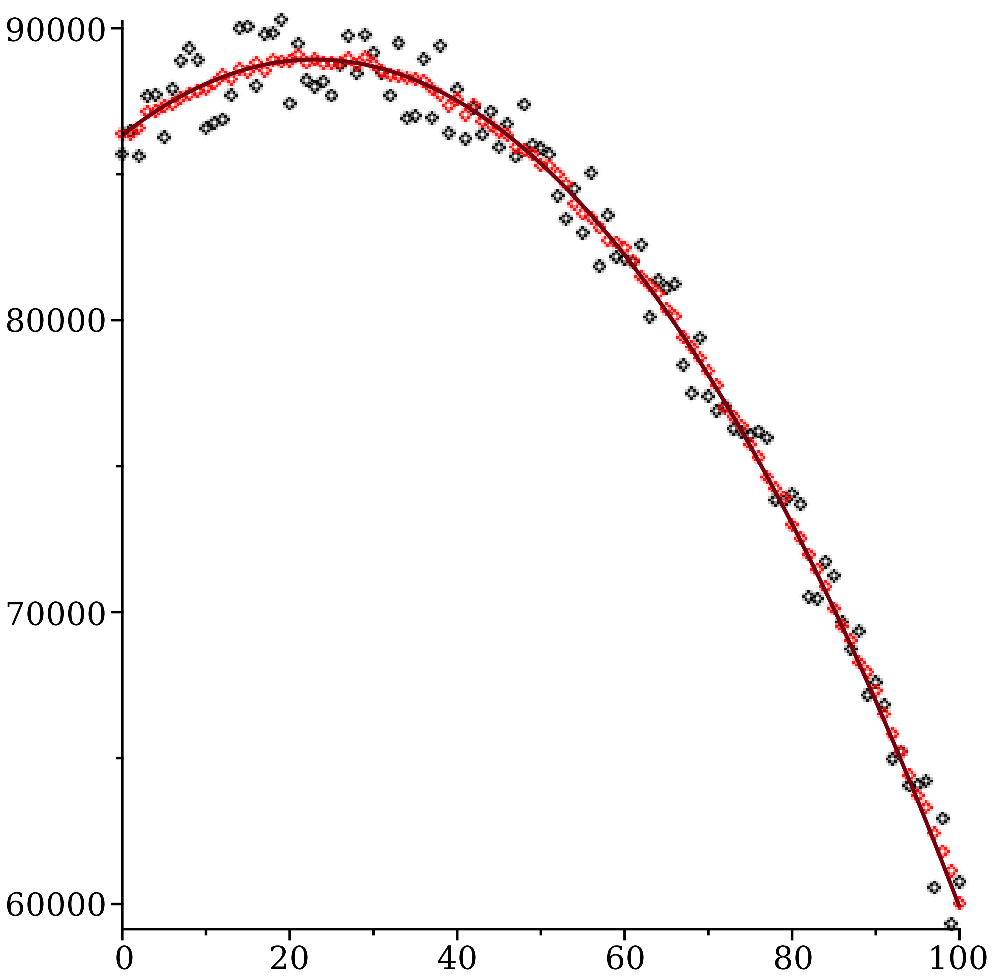


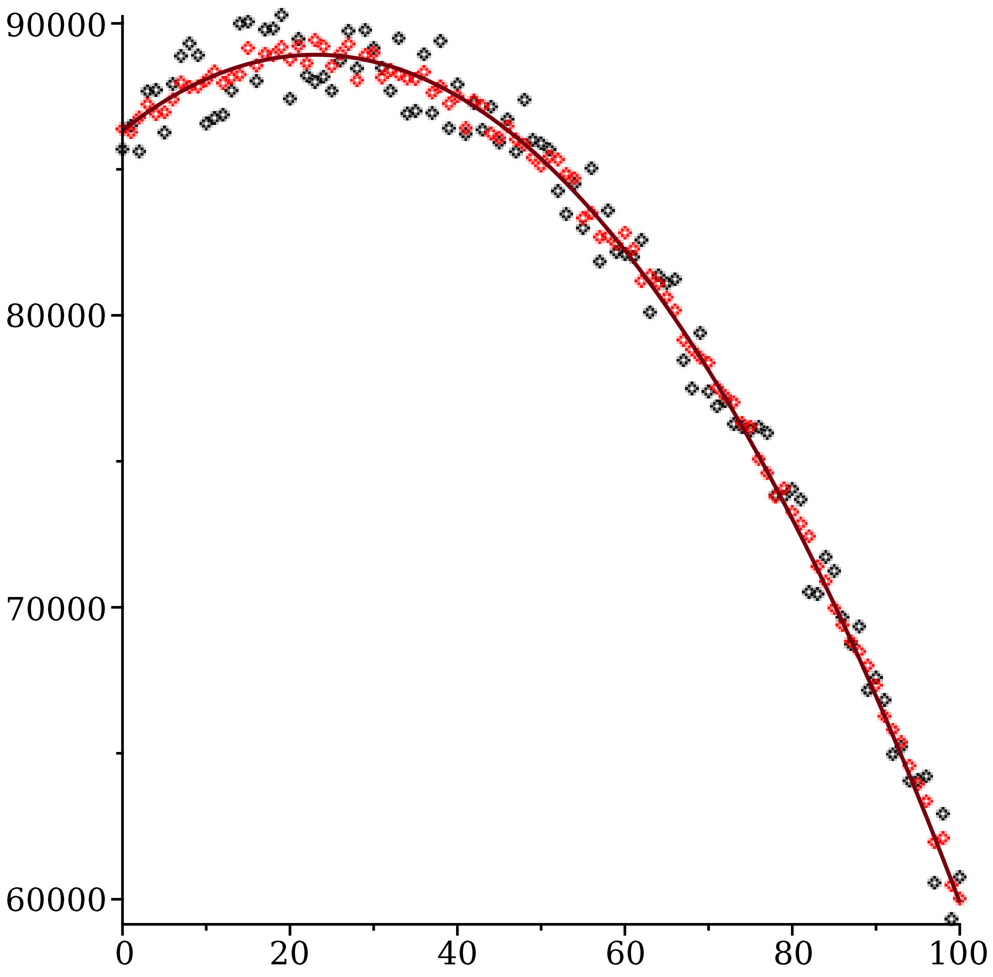
```
> i := 55;

  pointplot_i := pointplot(ts, M[..,i]):
  curveplot_i := plot(ts, R1[i]*map(f1,ts)+R2[i]*map(f2,ts)
  -9.8/2*map(f3,ts)):

  proj1 := (U[..,1].M[..,i]).U[..,1]:
  proj2 := (U[..,1].M[..,i]).U[..,1]+(U[..,2].M[..,i]).U[..,2]:
  proj3 := (U[..,1].M[..,i]).U[..,1]+(U[..,2].M[..,i]).U[..,2]+(U
  [..,3].M[..,i]).U[..,3]:
  proj4 := (U[..,1].M[..,i]).U[..,1]+(U[..,2].M[..,i]).U[..,2]+(U
  [..,3].M[..,i]).U[..,3]+(U[..,4].M[..,i]).U[..,4]:

  projectieplot1 := pointplot(ts, proj1, color=red):
  projectieplot2 := pointplot(ts, proj2, color=red):
  projectieplot3 := pointplot(ts, proj3, color=red):
  projectieplot4 := pointplot(ts, proj4, color=red):

  display(pointplot_i, projectieplot1);
  display(pointplot_i, projectieplot2);
  display(pointplot_i, projectieplot3, curveplot_i);
display(pointplot_i, projectieplot4, curveplot_i);
```

$i := 55$




90000
80000
70000
60000
0
20
40
60
80
100


90000
80000
70000
60000
0
20
40
60
80
100



Noisy oscillator

```
> restart:with(plots):with(LinearAlgebra):with(MTM):
```

Rijen en kolommen (# datapunten and # metingen)

```
> mmax := 100:
  nmax := 200:
```

Functies: cosinus en sinus

```
> f1 := t -> cos(0.2*t):
  f2 := t -> sin(0.2*t):
```

Tijdpunten (equidistant array/sequence)

```
> ts := [seq(t, t=0..mmax, 1)]:
```

Random functies: R voor random amplitudes, H voor random noise per meetpunt

```
> A  := 4.0:
  R  := rand(-A..A):
  H  := rand(-1.8..1.8):
```

lijst van amplitudes voor elke meting

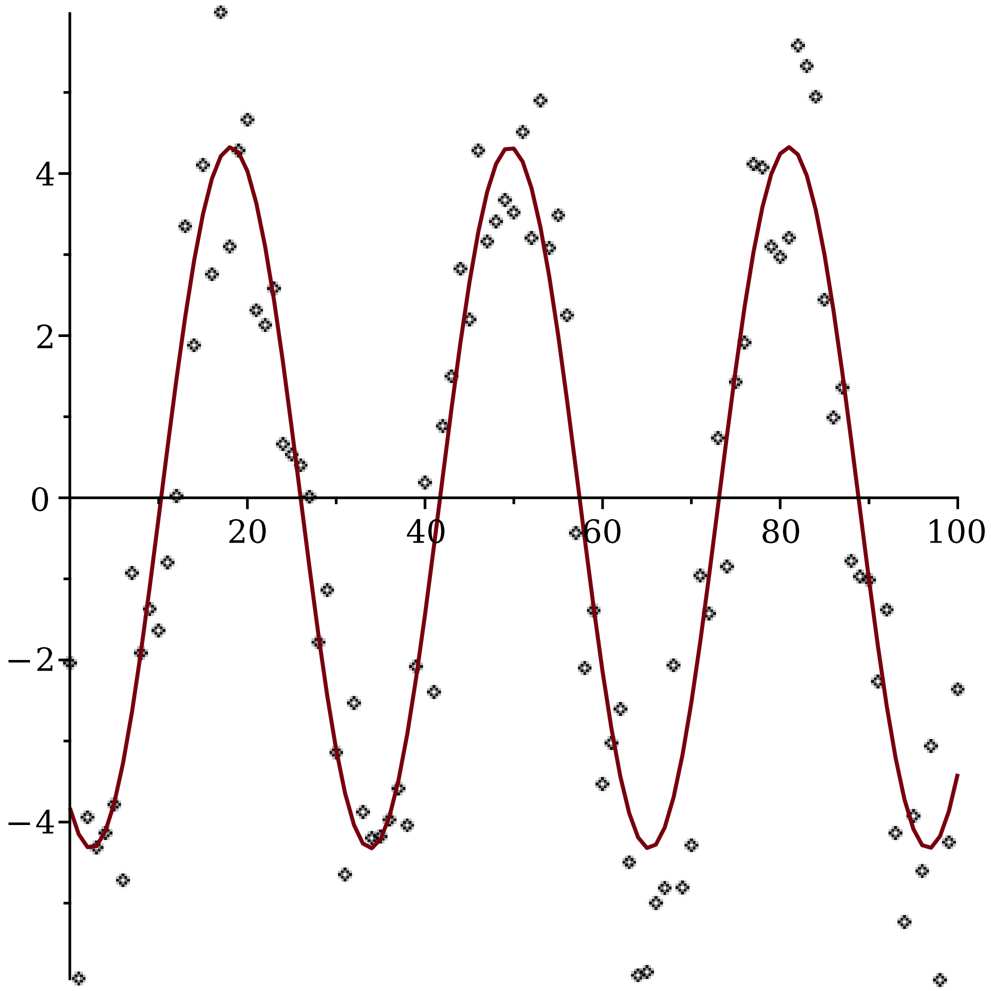
```
> R1 := [seq(R(), i=0..nmax)]:
  R2 := [seq(R(), i=0..nmax)]:
```

```
> M := Matrix(mmax+1, nmax, (i, j) -> R1[j]*f1(ts[i]) + R2[j]*f2
  (ts[i])+H()):
```

visualisatie meting i

```
> i := 30;
  pointplot_i := pointplot(ts, M[..,i]):
  curveplot_i := plot(ts, R1[i]*map(f1,ts)+R2[i]*map(f2,ts)):
  display(pointplot_i,curveplot_i);
```

$$i := 30$$



SVDecomposition

```
> U,S,V := svd(M):
```

Singular values: 2 grote (orde 250), rest zijn kleiner (orde 20)

```
> Diagonal(S);
```

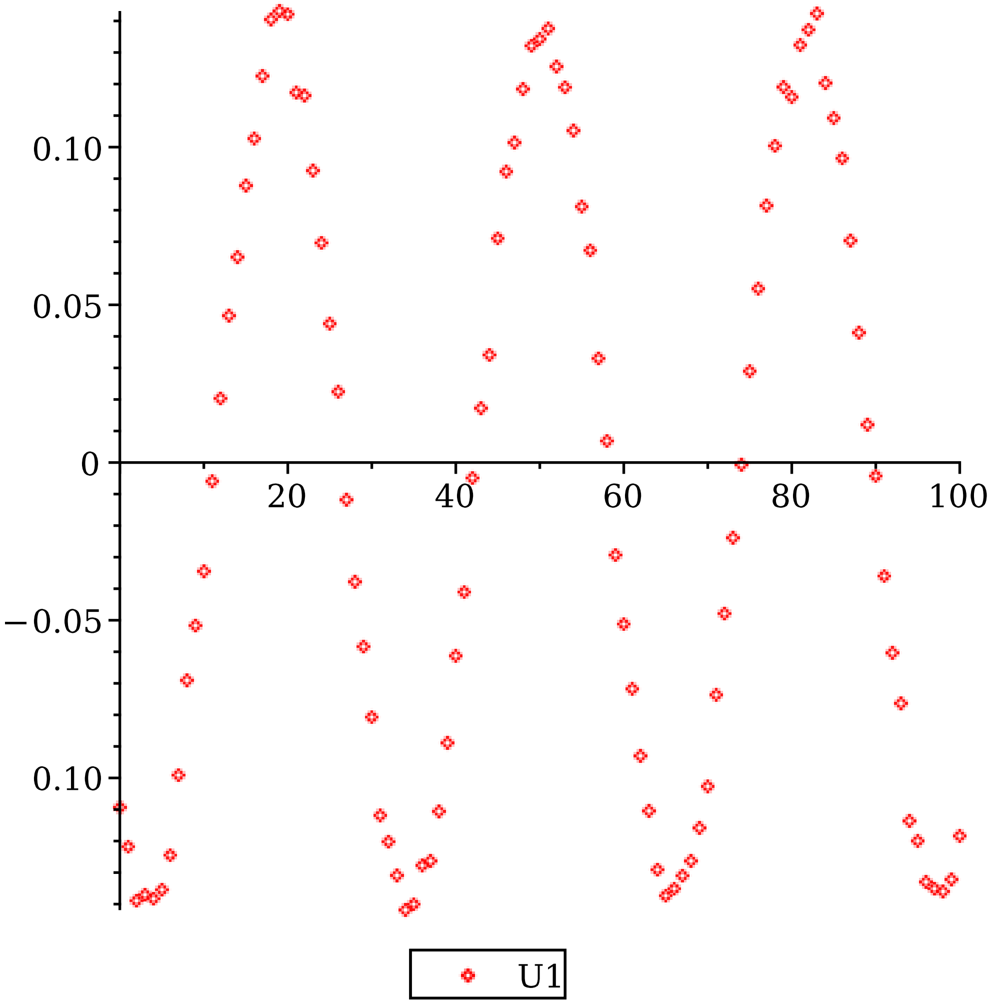
(4.2)

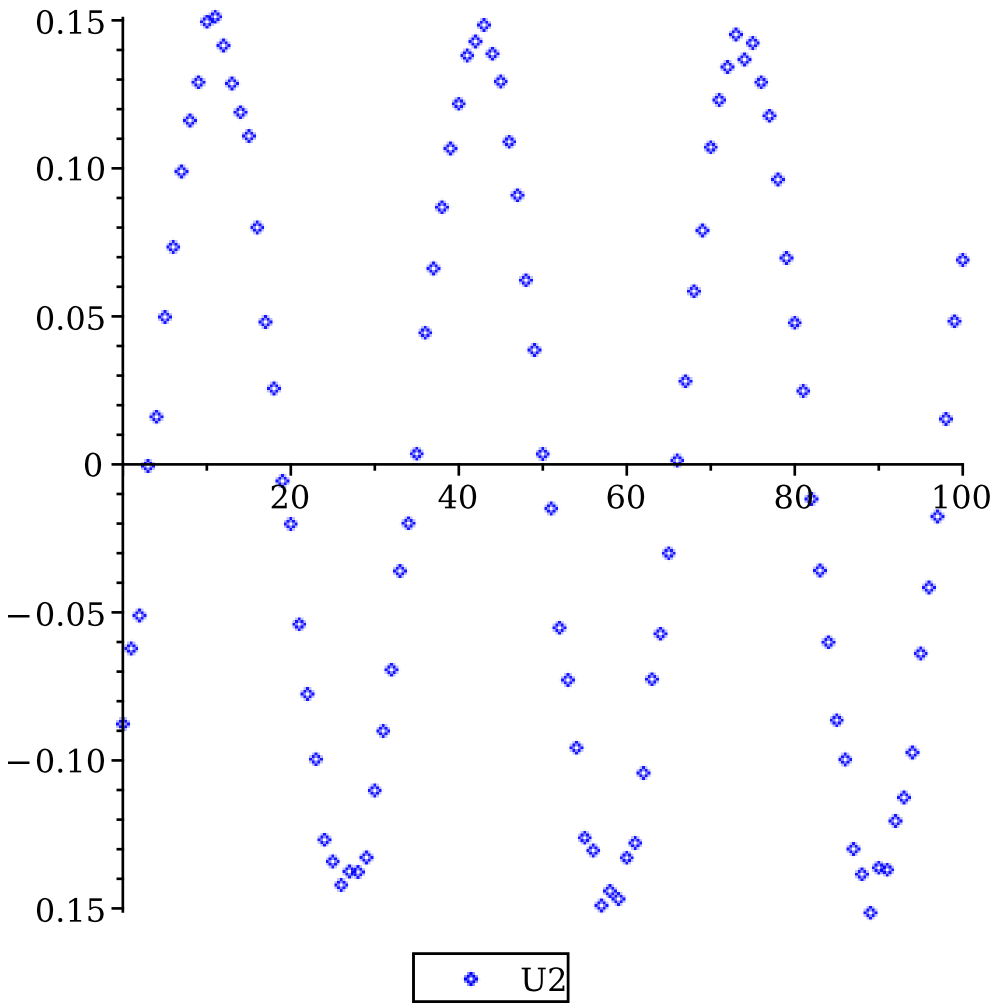
$$\begin{bmatrix} 255.451392358734 \\ 213.264462492594 \\ 25.1582279645861 \\ 24.0507702090590 \\ 23.9080041033373 \\ 23.1593145814381 \\ 22.9934092535045 \\ 22.1853405869249 \\ 21.8719264275831 \\ 21.6988494211531 \\ \vdots \end{bmatrix} \qquad (4.2)$$

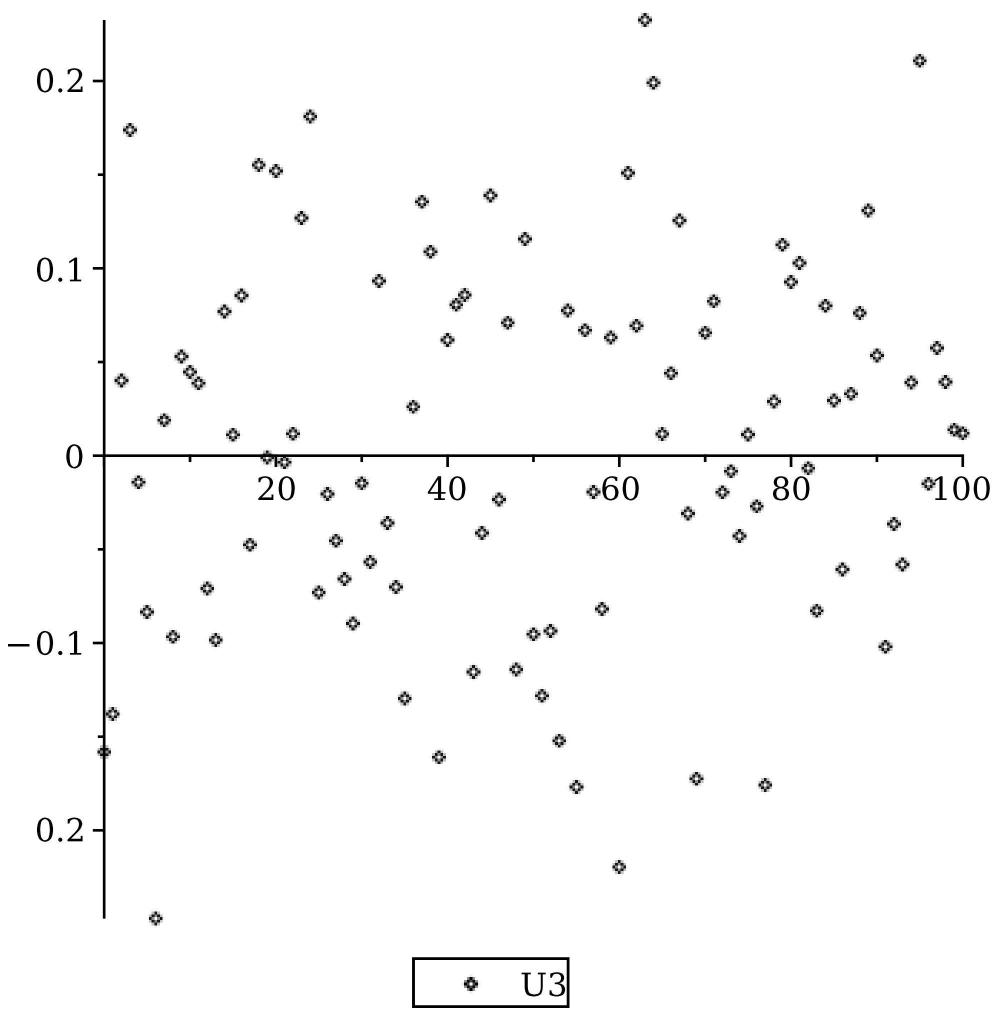
101 element Vector[column]

Plot U vectoren bij eerste 3 singular values

```
> U1plot := pointplot(ts, U[..,1], color=red, legend="U1");
  U2plot := pointplot(ts, U[..,2], color=blue, legend="U2");
  U3plot := pointplot(ts, U[..,3], color=black, legend="U3");
```


0.10
0.05
0
−0.05
0.10
20
40
60
80
100
U1


0.15
0.10
0.05
0
−0.05
−0.10
0.15
20
40
60
80
100
U2


0.2
0.1
0
−0.1
0.2
20
40
60
80
100
U3