Steierische Modellierungswoche 2012

Projekt: Signalverarbeitung

Tutorium: Trennung von Datenquellen in unkorrelierte und unabhängige Komponenten

a.o.Univ.Prof. Mag.Dr. Stephen Keeling
http://math.uni-graz.at/keeling/

Literatur:
http://cis.legacy.ics.tkk.fi/aapo/papers/IJCNN99_tutorialweb/

Dokumentation:
http://math.uni-graz.at/keeling/skripten/Tutorium.pdf

Dank an Herrn Dipl.-Ing. Dr. Gernot Reishofer
für seine Unterstützung für diese Arbeit!

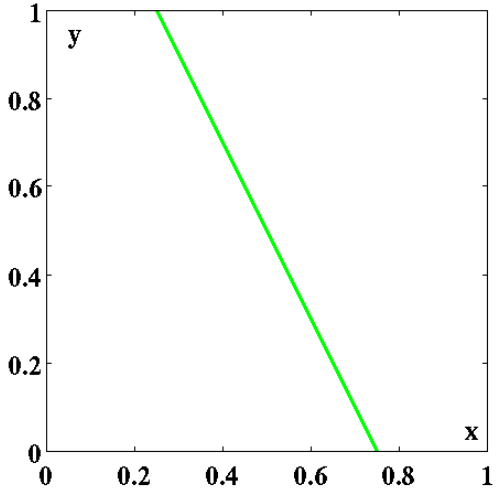
Inhaltsverzeichnis

Matrixalgebra

Beispiel einer linearen Gleichung:

$$4x + 2y = 3$$

Sie heisst *linear* wegen der grafischen Darstellung:



Mögliche Anwendung:
Die Nachfrage *y* fällt ab,
wenn der Preis *x* steigt.

Lineares Modell am einfachsten.

MATLAB Code für diese Grafik:

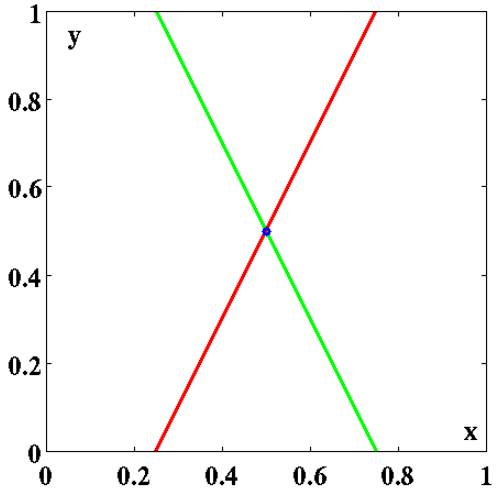
```
n = 101;
x = (0:(n-1))/(n-1);
y = (3-4*x)/2;
plot(x,y,'g');
```

Lineare Gleichungen

Beispiel einer zweiten linearen Gleichung:

$$4x + 2y = 3, \quad -4x + 2y = -1$$

Die beiden Gleichungen gelten im Schnittpunkt $(\frac{1}{2}, \frac{1}{2})$:



Ergänzte Anwendung:
Das Angebot y steigt,
wenn der Preis x steigt.

Kaufpreis $x = \frac{1}{2}$ entsteht natürlich.

MATLAB Code für diese Grafik:

```
n = 101;
x = linspace(0,1,n); xP = 0.5;
yN = (3-4*x)/2; yA = (-1+4*x)/2; yP = 0.5;
plot(x,yN,'g',x,yA,'r',xP,yP,'bo');
```

Lösung von Systemen Linearer Gleichungen

Wie findet man den Schnittpunkt?

$$\begin{array}{rrcll} & 4x+2y & = & 3 & (\mathrm{I}) \\ & -4x+2y & = & -1 & (\mathrm{II}) \\ \hline (\mathrm{I})+(\mathrm{II})\rightarrow & 4y & = & 2 & \Rightarrow y=\frac{1}{2} \\ \hline (\mathrm{I})\rightarrow & 4x+2(\frac{1}{2}) & = & 3 & \Rightarrow x=\frac{1}{2} \end{array}$$

Allgemeiner:

$$\begin{array}{rrcll} & a_{11}x_1+a_{12}x_2 & = & b_1 & (\mathrm{I}) \\ m_{21}=a_{11}/a_{21} & a_{21}x_1+a_{22}x_2 & = & b_2 & (\mathrm{II}) \\ \hline -m_{21}(\mathrm{II})\rightarrow & -a_{11}x_1-m_{21}a_{22}x_2 & = & -m_{21}b_2 & (\mathrm{III}) \\ \hline (\mathrm{I})+(\mathrm{III})\rightarrow & (a_{12}-m_{21}a_{22})x_2 & = & (b_1-m_{21}b_2) & \\ & & & \Rightarrow \cdots x_2=\frac{a_{11}b_2-a_{21}b_1}{a_{11}a_{22}-a_{21}a_{12}} & \\ \hline (\mathrm{I})\rightarrow & a_{11}x_1+a_{12}\frac{a_{11}b_2-a_{21}b_1}{a_{11}a_{22}-a_{21}a_{12}} & = & b_1 & \\ & & & \Rightarrow x_1=\frac{b_1a_{22}-b_2a_{12}}{a_{11}a_{22}-a_{21}a_{12}} & \end{array}$$

Bemerke die notwendige Bedingung:

$$\det[A]=a_{11}a_{22}-a_{21}a_{12}\neq 0$$

Lösung von Systemen Linearer Gleichungen

MATLAB Code zur Lösung dieses Systems:

```
A = [4,2;-4,2];
b = [3;-1];
x = A\b;
```

Gespeichert in `probe.m`, ausgeführt im Befehlsfenster:

```
>> probe
>> x
x =
   0.5000
   0.5000
```

Lösung vom 3×3 System, $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$,

$$\begin{bmatrix} 4 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \left\{ \begin{array}{rcrcrcl} 4x_1 & + & 2x_2 & + & x_3 & = & 1 \\ 2x_1 & + & 4x_2 & + & 2x_3 & = & 2 \\ x_1 & + & 2x_2 & + & 4x_3 & = & 3 \end{array} \right\} \leftarrow \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

oder $\boldsymbol{x} = A^{-1}\boldsymbol{b}$, läuft analog mit MATLAB:

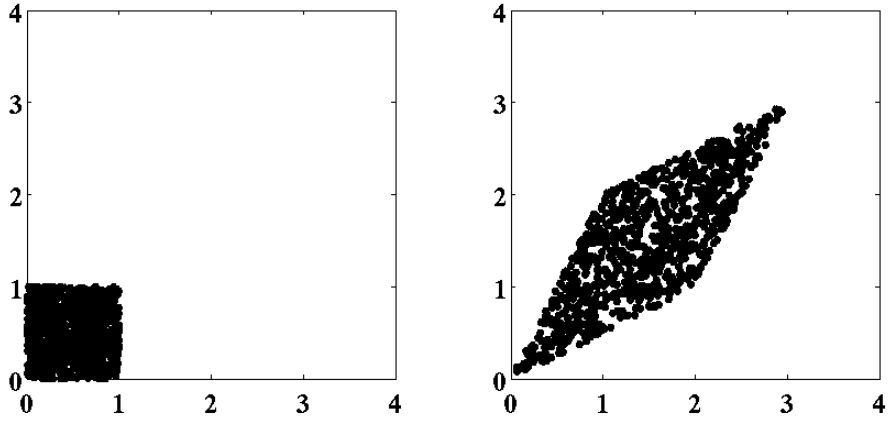
```
A = [4,2,1;2,4,2;1,2,4]; b = [1;2;3]; x = A\b;
```

Effekt der Matrix-Multiplikation

Für die folgende Matrix A werden alle Punkte $\boldsymbol{x} = \langle x_1, x_2 \rangle^{\mathrm{T}}$ (links) zu Punkte $A\boldsymbol{x}$ (rechts) abgebildet:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

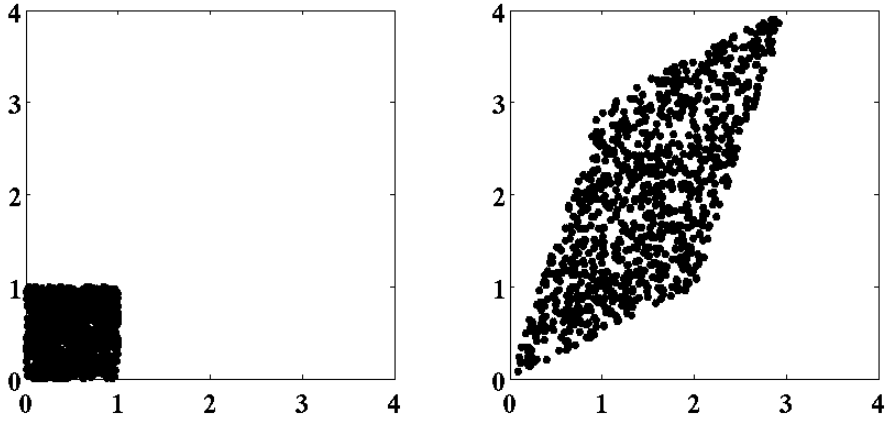
(symmetrisch)



Für die folgende Matrix A werden alle Punkte $\boldsymbol{x}$ (links) zu Punkte $A\boldsymbol{x}$ (rechts) abgebildet:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$$

(nicht symmetrisch)



Effekt der Matrix-Multiplikation

MATLAB Code für diese Grafiken:

```
A = [2,1;1,2];
```

oder

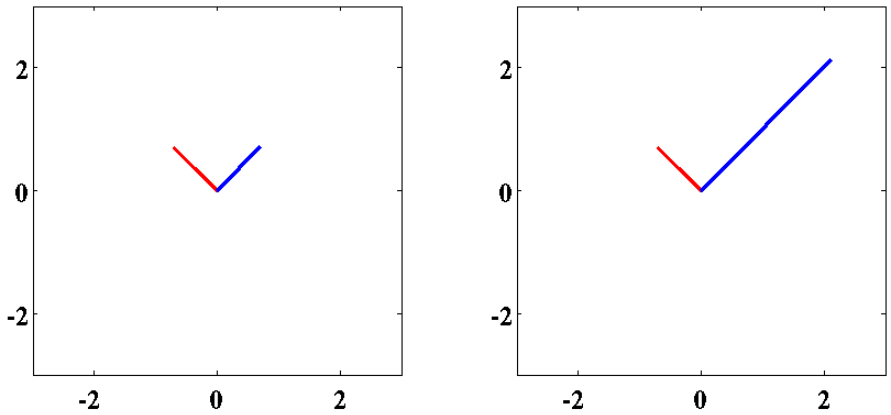
```
A = [1,2;3,1];

n = 1001;
x = rand(2,n);
y = A*x;

subplot(1,2,1)
plot(x(1,:),x(2,:),'k*')
axis([0 4 0 4]);

subplot(1,2,2)
plot(y(1,:),y(2,:),'k*')
axis([0 4 0 4]);
```

Effekt der Matrix-Multiplikation

Für die folgende Matrix A werden alle Punkte $\boldsymbol{x}$ (links) zu Punkte $A\boldsymbol{x}$ (rechts) abgebildet:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

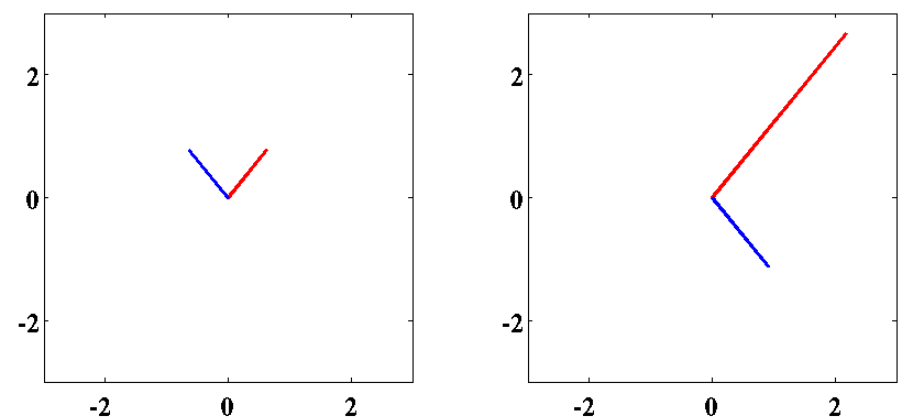
(symmetrisch)



Für die folgende Matrix A werden alle Punkte $\boldsymbol{x}$ (links) zu Punkte $A\boldsymbol{x}$ (rechts) abgebildet:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$$

(nicht symmetrisch)



Eigenräume

Wie findet man solche *Eigenräume*? Durch Lösung einer Gleichung,

$$A\boldsymbol{v} = \lambda \boldsymbol{v}$$

Beispiel: Mit der ersten Matrix A oben,

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \lambda \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \quad \Leftrightarrow \begin{cases} (2-\lambda)v_1 + v_2 &= 0 \\ v_1 + (2-\lambda)v_2 &= 0 \end{cases}$$

Mit den obigen Lösungsformeln ergibt sich $v_1 = v_2 = 0$ ausser

$$\det \begin{bmatrix} (2-\lambda) & 1 \\ 1 & (2-\lambda) \end{bmatrix} = (2-\lambda)^2 - 1 = 0 \quad \text{oder} \quad \lambda \in \{1, 3\}.$$

Insbesondere

$$\lambda = 1 \Rightarrow \begin{cases} v_1 + v_2 &= 0 \\ v_1 + v_2 &= 0 \end{cases} \qquad \lambda = 3 \Rightarrow \begin{cases} -v_1 + v_2 &= 0 \\ v_1 - v_2 &= 0 \end{cases}$$

oder $\lambda = 1 \Rightarrow v_1 = -v_2$ und $\lambda = 3 \Rightarrow v_1 = v_2$. Siehe Grafik!

Eigenwerte und Eigenvektoren

Beispiel: Mit der zweiten Matrix A oben,

$$\begin{bmatrix} 1 & 3 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \lambda \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \quad \Leftrightarrow \begin{cases} (1-\lambda)v_1 + 3v_2 &= 0 \\ 2v_1 + (1-\lambda)v_2 &= 0 \end{cases}$$

Die triviale Lösung $v_1 = v_2 = 0$ ergibt sich ausser

$$\det \begin{bmatrix} (1-\lambda) & 3 \\ 2 & (1-\lambda) \end{bmatrix} = (1-\lambda)^2 - 6 = 0 \quad \text{oder} \quad \lambda \in \{1-\sqrt{6}, 1+\sqrt{6}\}$$

wobei $1-\sqrt{6} \approx -1.45$ und $1+\sqrt{6} \approx 3.45$. Insbesondere

$$\lambda = 1-\sqrt{6} \Rightarrow \begin{cases} \sqrt{6}v_1 + 3v_2 &= 0 \\ 2v_1 + \sqrt{6}v_2 &= 0 \end{cases} \qquad \lambda = 1+\sqrt{6} \Rightarrow \begin{cases} -\sqrt{6}v_1 + 3v_2 \\ 2v_1 - \sqrt{6}v_2 \end{cases}$$

oder die *Eigenwerte*

$$\lambda_1 = 1-\sqrt{6} \quad \text{und} \quad \lambda_2 = 1+\sqrt{6}.$$

haben die entsprechenden Lösungen oder *Eigenvektoren*

$$\boldsymbol{v}_1 = \langle -\sqrt{\tfrac{3}{5}}, \sqrt{\tfrac{2}{5}} \rangle^{\mathrm{T}} \quad \text{und} \quad \boldsymbol{v}_2 = \langle \sqrt{\tfrac{3}{5}}, \sqrt{\tfrac{2}{5}} \rangle^{\mathrm{T}}$$

mit der üblichen *Normalisierung* $\|\boldsymbol{v}_1\| = \|\boldsymbol{v}_2\| = 1$. Siehe Grafik!

Eigenraum-Zerlegung

Seien die Eigenräume gegeben:

$$\boldsymbol{A}\boldsymbol{v}_1 = \boldsymbol{v}_1\lambda_1, \quad \boldsymbol{A}\boldsymbol{v}_2 = \boldsymbol{v}_2\lambda_2$$

Mit den Matrizen,

$$V = \{\boldsymbol{v}_1, \boldsymbol{v}_2\}, \quad \Lambda = \mathrm{diag}\{\lambda_1, \lambda_2\}$$

kann die Eigenraum-Zerlegung so umgeschrieben werden:

$$AV = V\Lambda$$

Ein beliebiger Vektor $\boldsymbol{v}$ kann durch die Lösung,

$$\boldsymbol{v} = V\boldsymbol{z}, \quad \boldsymbol{z} = V^{-1}\boldsymbol{v}, \quad \boldsymbol{z} = \langle z_1, z_2\rangle^{\mathrm{T}}$$

bezüglich der neuen Basis $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ dargestellt werden,

$$\boldsymbol{v} = z_1\boldsymbol{v}_1 + z_2\boldsymbol{v}_2$$

So wird die Wirkung der Matrix A,

$$\boldsymbol{A}\boldsymbol{v} = z_1\boldsymbol{A}\boldsymbol{v}_1 + z_2\boldsymbol{A}\boldsymbol{v}_2 = z_1\lambda_1\boldsymbol{v}_1 + z_2\lambda_2\boldsymbol{v}_2$$

d.h.

$$\boldsymbol{A}\boldsymbol{v} = (V\Lambda V^{-1})(V\boldsymbol{z}) = V(\Lambda\boldsymbol{z})$$

entkoppelt in 1D Wirkungen auf jeder Achse $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ eines neuen *verzerrten* Achsensystems. Siehe Grafik!

Eigenraum-Zerlegung

Wie oben erklärt, werden die Eigenvektoren normalisiert:

$$\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{v}_1 = \boldsymbol{v}_2^{\mathrm{T}}\boldsymbol{v}_2 = 1$$

Wenn A symmetrisch ist, sind die Eigenvektoren senkrecht auf einander!

$$\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{v}_2 = 0$$

Diese Bedingungen können mit Matrizen so umgeschrieben werden:

$$V^{\mathrm{T}}V = I, \quad V^{-1} = V^{\mathrm{T}}$$

So ist V eine *Drehungsmatrix*. Folgendes Umschreiben entspricht einer Drehung:

$$\boldsymbol{v} = V\boldsymbol{d}, \quad \boldsymbol{d} = V^{\mathrm{T}}\boldsymbol{v}$$

So wird die Wirkung der Matrix A,

$$A\boldsymbol{v} = (V\Lambda V^{\mathrm{T}})(V\boldsymbol{d}) = V(\Lambda\boldsymbol{d})$$

entkoppelt in 1D Wirkungen auf jeder Achse $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ eines neuen *ge**d**rehten* Achsensystems. Siehe Grafik!

Eigenraum-Zerlegung

MATLAB Code zur Bestimmung der Eigenräume:
Im Befehlsfenster,

```
>> A = [1,3;2,1];
>> [V,L] = eig(A)
   V =
      0.7746 -0.7746
      0.6325  0.6325
   L =
      3.4495       0
           0 -1.4495
>> inv(V)*A*V
ans =
      3.4495  0.0000
     -0.0000 -1.4495
```

```
>> A = [2,1;1,2];
>> [V,L] = eig(A)
   V =
     -0.7071 0.7071
      0.7071 0.7071
   L =
      1 0
      0 3
>> V'*A*V
ans =
   1.0000       0
        0 3.0000
```

Vergleiche mit den obigen händischen Rechnungen.

Eigenraum-Zerlegung

Beispiel: Für einen gegeben Vektor

$$\boldsymbol{v} = n_1 \boldsymbol{e}_1 + n_2 \boldsymbol{e}_2$$

wobei

$$\boldsymbol{e}_1 = \langle 1, 0 \rangle^{\mathrm{T}} \text{ und } \boldsymbol{e}_2 = \langle 0, 1 \rangle^{\mathrm{T}},$$

wie bestimmt man in dem ***n**atürlichen* Achsensystem $\{\boldsymbol{e}_1, \boldsymbol{e}_2\}$ die Koordinaten $\boldsymbol{n} = \langle n_1, n_2 \rangle^{\mathrm{T}}$? Methode:

$$\boldsymbol{e}_1^{\mathrm{T}} \boldsymbol{v} = n_1 (\boldsymbol{e}_1^{\mathrm{T}} \boldsymbol{e}_1)_{=1} + n_2 (\boldsymbol{e}_1^{\mathrm{T}} \boldsymbol{e}_2)_{=0} = n_1$$
$$\boldsymbol{e}_2^{\mathrm{T}} \boldsymbol{v} = n_1 (\boldsymbol{e}_2^{\mathrm{T}} \boldsymbol{e}_1)_{=0} + n_2 (\boldsymbol{e}_2^{\mathrm{T}} \boldsymbol{e}_2)_{=1} = n_2$$

oder mit $I = \{\boldsymbol{e}_1, \boldsymbol{e}_2\}$,

$$I^{\mathrm{T}} \boldsymbol{v} = \boldsymbol{n}.$$

Wie bestimmt man die Koordinaten $\boldsymbol{d} = \langle d_1, d_2 \rangle^{\mathrm{T}}$

$$\boldsymbol{v} = d_1 \boldsymbol{v}_1 + d_2 \boldsymbol{v}_2$$

in dem *ge**d**rehten* Achsensystem der Eigenvektoren $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ der Matrix $A = [2, 1; 1, 2]$? Methode:

$$\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{v} = d_1 (\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{v}_1)_{=1} + d_2 (\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{v}_2)_{=0} = d_1$$
$$\boldsymbol{v}_2^{\mathrm{T}} \boldsymbol{v} = d_1 (\boldsymbol{v}_2^{\mathrm{T}} \boldsymbol{v}_1)_{=0} + d_2 (\boldsymbol{v}_2^{\mathrm{T}} \boldsymbol{v}_2)_{=1} = d_2$$

oder mit $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2\}$,

$$V^{\mathrm{T}} \boldsymbol{v} = \boldsymbol{d}.$$

Eigenraum-Zerlegung

Beispiel: Wie bestimmt man die Koordinaten $\boldsymbol{z} = \langle z_1, z_2 \rangle^{\mathrm{T}}$

$$\boldsymbol{v} = z_1 \boldsymbol{v}_1 + z_2 \boldsymbol{v}_2$$

in dem *verzerrten* Achsensystem der Eigenvektoren $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ der Matrix $A = [1, 3; 2, 1]$? Methode:

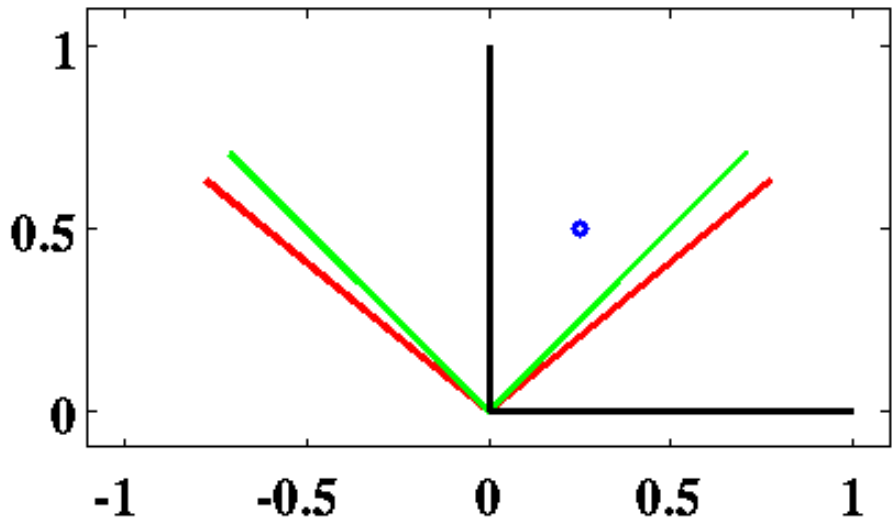
$$\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{v} = z_1 (\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{v}_1)_{=1} + z_2 (\boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{v}_2)_{\neq 0} = z_1 + (?) z_2$$
$$\boldsymbol{v}_2^{\mathrm{T}} \boldsymbol{v} = z_1 (\boldsymbol{v}_2^{\mathrm{T}} \boldsymbol{v}_1)_{\neq 0} + z_2 (\boldsymbol{v}_2^{\mathrm{T}} \boldsymbol{v}_2)_{=1} = (?) z_1 + z_2$$

Mit $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2\}$, müssen die Koordinaten so berechnet werden:

$$\boldsymbol{v} = V\boldsymbol{z}, \quad \boldsymbol{z} = V^{-1}\boldsymbol{v}.$$

Beispiel:

Für den Punkt ***v***, finde die ***n****atürlichen* Koordinaten ***n***, die Koordinaten ***d*** des *ge****d****rehten* Eigensystems von $A = [2, 1; 1, 2]$ und die Koordinaten ***z*** des *ver****z****errten* Eigensystems von $A = [1, 3; 2, 1]$.



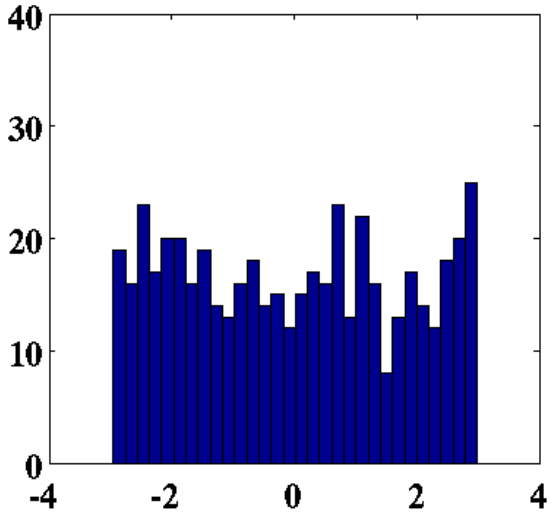
Statistik

Sei $\boldsymbol{x} = \{x_i\}_{i=1}^n$ eine Abtastung.

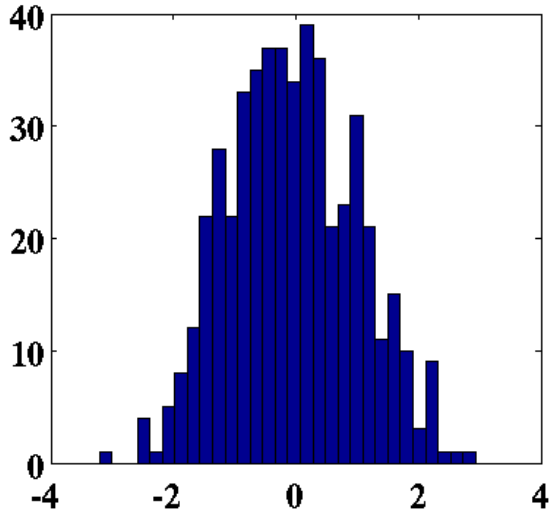
Der *Mittelwert* ist: $$\mu(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Die *Standardabweichung* ist $\sigma(\boldsymbol{x})$ und die *Varianz* ist: $$\sigma^2(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu(\boldsymbol{x}))^2$$

Beispiele: Gleichmäßig und Gauß verteilte Abtastungen.



$\mu(\boldsymbol{x}) = -0.0214,$
$\sigma^2(\boldsymbol{x}) = 3.1860$



$\mu(\boldsymbol{y}) = -0.0626,$
$\sigma^2(\boldsymbol{y}) = 1.0834$

Histogramm: Anzahl der Werte $\{x_i\}$ in einem diskreten *bin* vs. bin.

Mittelwert und Varianz einer Abtastung

MATLAB Code für diese Grafiken:

```
n = 501;
x = rand(1,n);   % gleichmaessig in [0,1]
x = 6*(x-0.5);   % gleichmaessig in [-3,3]
disp(strcat('x-Mittelwert=',num2str(mean(x))), ...
     ' x-Varianz=',num2str(std(x)^2);
y = randn(1,n); % Gauss (normal)
disp(strcat('y-Mittelwert=',num2str(mean(y))), ...
     ' y-Varianz=',num2str(std(y)^2);
subplot(1,2,1);
hist(x,30);      % x-Histogramm mit 30 bins
axis([-4 4 0 40]);
subplot(1,2,2);
hist(y,30);      % y-Histogramm mit 30 bins
axis([-4 4 0 40]);
```

Beispiel: Wie ändert man $\mu(\boldsymbol{y})$ und $\sigma(\boldsymbol{y})$?

Zentraler Grenzwertsatz

Warum ist die Gaußsche Verteilung so wichtig?

Zentraler Grenzwertsatz: Sei $\{x_i\}_{i=1}^n$ eine Folge von unabhängigen identisch verteilten Zufallsvariablen mit (theoretischen) Mittelwert μ und Standardabweichung σ. Mit

$$\overline{x}_n = \frac{1}{n}\sum_{i=1}^{n} x_i$$

sei Φ_n die Verteilung der Zufallsvariable

$$\frac{\overline{x}_n - \mu}{\sigma/\sqrt{n}}$$

Sei Φ die Verteilung einer Gaußschen (normalen) Zufallsvariable $N(0,1)$ mit Mittelwert 0 und Standardabweichung 1. Dann gilt

$$\Phi_n \overset{n\to\infty}{\longrightarrow} \Phi.$$

Praktische Konsequenz: Eine lineare Mischung (Summe) is *normaler* als die Quellen! (Rückgängig machen?)

Beispiel: Sogar für $x \sim U(\mu,\sigma)$ (gleichmäßig), gilt $\frac{\overline{x}_n-\mu}{\sigma/\sqrt{n}} \overset{n\to\infty}{\longrightarrow} N(0,1)$.

Kovarianz zweier Abtastungen

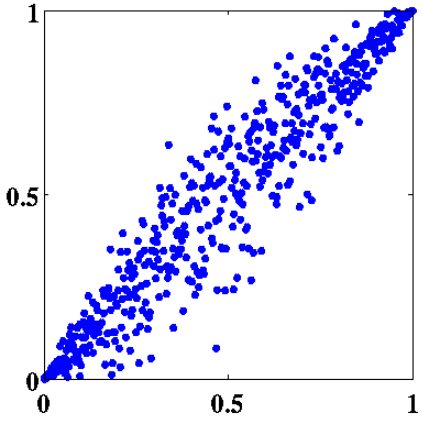
Die Varianz $\sigma^2(\boldsymbol{x})$ für $\boldsymbol{x} = \{x_i\}_{i=1}^n$ lässt sich so umschreiben:

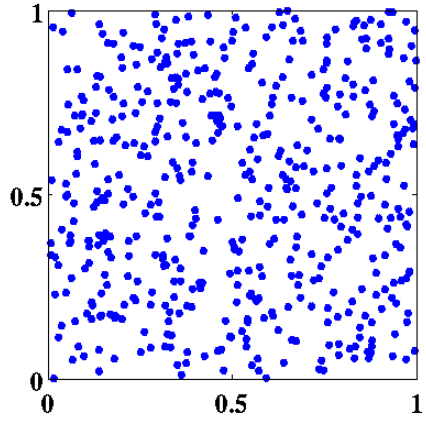
$$\sigma^2(\boldsymbol{x}) = \frac{1}{n}[\boldsymbol{x} - \mu(\boldsymbol{x})]^{\mathrm{T}}[\boldsymbol{x} - \mu(\boldsymbol{x})]$$

Analog gibt es eine *Kovarianz* zwischen $\boldsymbol{x} = \{x_i\}_{i=1}^n$ und $\boldsymbol{y} = \{y_i\}_{i=1}^n$,

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n}[\boldsymbol{x} - \mu(\boldsymbol{x})]^{\mathrm{T}}[\boldsymbol{y} - \mu(\boldsymbol{y})]$$

Beispiel:





$\kappa(\boldsymbol{x}, \boldsymbol{x}) = 0.0837$
$\kappa(\boldsymbol{x}, \boldsymbol{y}) = 0.0831$
$\kappa(\boldsymbol{y}, \boldsymbol{y}) = 0.0910$

$\kappa(\boldsymbol{x}, \boldsymbol{x}) = 0.0844$
$\kappa(\boldsymbol{x}, \boldsymbol{y}) = 0.0026$
$\kappa(\boldsymbol{y}, \boldsymbol{y}) = 0.0789$

Kovarianzmatrix:

$$K = \begin{bmatrix} \kappa(\boldsymbol{x}, \boldsymbol{x}) & \kappa(\boldsymbol{x}, \boldsymbol{y}) \\ \kappa(\boldsymbol{y}, \boldsymbol{x}) & \kappa(\boldsymbol{y}, \boldsymbol{y}) \end{bmatrix}$$

Je höher ist $\kappa(\boldsymbol{x}, \boldsymbol{y})$, desto enger liegen die Punkte in Richtung des Eigenvektors von K mit dem größten Eigenwert. Je kleiner, desto ähnlicher die Eigenwerte.

Kovarianz zweier Abtastungen

MATLAB Code für diese Grafik:

```
n = 501;

x = linspace(0,1,n);
y = x + 0.5*x.*(1-x).*randn(1,n);
subplot(1,2,1)
plot(x,y,'b*');
mux = mean(x); muy = mean(y);
[(x-mux)*(x-mux)',(x-mux)*(y-muy)',(y-muy)*(y-muy)']/n

x = rand(1,n);
y = rand(1,n);
subplot(1,2,2)
plot(x,y,'b*');
mux = mean(x); muy = mean(y);
[(x-mux)*(x-mux)',(x-mux)*(y-muy)',(y-muy)*(y-muy)']/n
```

Zentrierte und Gesphärte Daten

Wie können die Daten transformiert werden, damit die Kovarianz reduziert wird? Definiere die *zentrierten* Daten,

$$Y_z = \begin{bmatrix} \boldsymbol{y}_1^T - \mu(\boldsymbol{y}_1) \\ \boldsymbol{y}_2^T - \mu(\boldsymbol{y}_2) \end{bmatrix} \in \mathbb{R}^{2\times n}$$

um die Kovarianzmatrix so umzuschreiben:

$$K = \tfrac{1}{n} Y_z Y_z^T \in \mathbb{R}^{2\times 2}$$

Diese Matrix ist *symmetrisch und positiv semidefinit*:

$$\boldsymbol{v}^T Y_z Y_z^T \boldsymbol{v} = (Y_z^T \boldsymbol{v})^T (Y_z^T \boldsymbol{v}) = \| Y_z^T \boldsymbol{v}\|^2 \geq 0, \quad 0 \neq \boldsymbol{v} \in \mathbb{R}^2$$

Solche Matrizen haben Eigenvektoren mit $\boldsymbol{v}_i^T \boldsymbol{v}_j \in \{0, 1\}$ und nicht negative Eigenwerte $\lambda_i \geq 0$. Dann ist $\Lambda^{-\frac{1}{2}}$ wohl definiert ▸.

Mit der Eigenraum-Zerlegung,

$$KV = V\Lambda \quad \Rightarrow \quad V\Lambda V^T = \tfrac{1}{n} Y_z Y_z^T \quad \Rightarrow \quad \Lambda = \tfrac{1}{n} V^T Y_z Y_z^T V$$

folgt

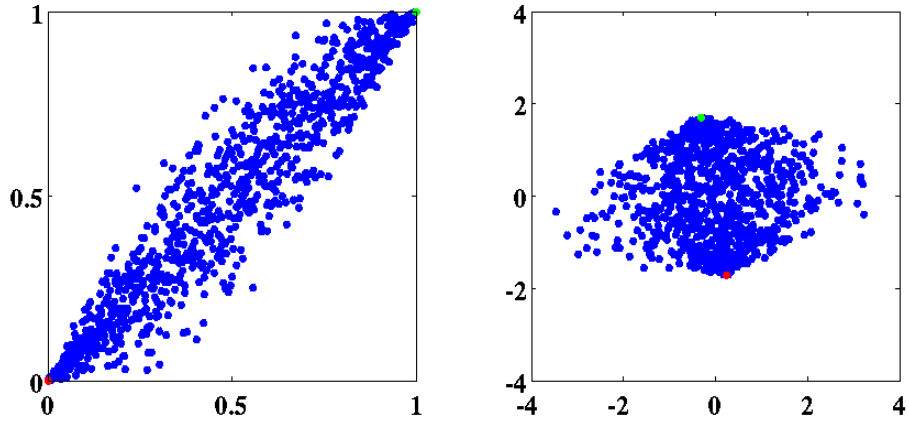
$$I = \Lambda^{-\frac{1}{2}} \Lambda \Lambda^{-\frac{1}{2}} = \tfrac{1}{n} \Lambda^{-\frac{1}{2}} V^T Y_z Y_z^T V \Lambda^{-\frac{1}{2}} = \tfrac{1}{n} (Y_z^T V \Lambda^{-\frac{1}{2}})^T (Y_z^T V \Lambda^{-\frac{1}{2}})$$

Daher sind die *gesphärten* Daten

$$Y_s = \Lambda^{-\frac{1}{2}} V^T Y_z \quad \text{unkorreliert,} \quad \tfrac{1}{n} Y_s Y_s^T = I.$$

Zentrierte und Gesphärte Daten

Die Daten werden mit der obigen Formel so gesphärt:



MATLAB Code für diese Grafik:

```
n = 1001; Y = zeros(2,n); Yz = zeros(2,n);
Y(1,:)=linspace(0,1,n);
Y(2,:)=Y(1,:)+0.5*Y(1,:).*(1-Y(1,:)).*randn(1,n);
subplot(1,2,1); plot(Y(1,:),Y(2,:),'b*');
Yz(1,:)  = Y(1,:)  - mean(Y(1,:));
Yz(2,:)  = Y(2,:)  - mean(Y(2,:));
K = Yz*Yz'/n; [V,D] = eig(K);
Ys = diag(1./sqrt(diag(D)))*V'*Yz;
subplot(1,2,2); plot(Ys(1,:),Ys(2,:),'b*');
```

Kovarianz mehrerer Abtastungen

Seien m Abtastungen $\{\boldsymbol{y}_k\}_{k=1}^m$ gegeben, und definiere

$$Y_z = \begin{bmatrix} \boldsymbol{y}_1^T - \mu(\boldsymbol{y}_1) \\ \vdots \\ \boldsymbol{y}_m^T - \mu(\boldsymbol{y}_m) \end{bmatrix} \in \mathbb{R}^{m \times n}$$

Die Kovarianzmatrix ist:

$$K = \frac{1}{n} Y_z Y_z^T \in \mathbb{R}^{m \times m}$$

Aus der Eigenraum-Zerlegung $KV = V\Lambda$ folgt es wie vorher, dass $Y_s = \Lambda^{-\frac{1}{2}} V^T Y_z$ die Kovarianz $\frac{1}{n} Y_s Y_s^T = I$ hat.

Wenn $m \gg n$ gilt (z.B. # Pixel $\gg$ # Zeitpunkte), wird die obige Zerlegung leichter durch folgende Eigenraum-Zerlegung bestimmt:

$$Y_z^T Y_z = \hat{Y}_s \hat{\Lambda} \hat{Y}_s^T \in \mathbb{R}^{n \times n} \text{ wobei } \Lambda = \begin{bmatrix} \hat{\Lambda} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{m \times m},\ Y_s = \begin{bmatrix} \hat{Y}_s \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^m$$

und $\frac{1}{n} \hat{Y}_s^T \hat{Y}_s = I$. Dann mit $\hat{V} = Y_z \hat{Y}_s \hat{\Lambda}^{-\frac{1}{2}} \in \mathbb{R}^{m \times n}$ folgt

$K\hat{V} = \frac{1}{n} Y_z (Y_z^T Y_z) \hat{Y}_s \hat{\Lambda}^{-\frac{1}{2}} = \frac{1}{n} Y_z (\hat{Y}_s \hat{\Lambda} \hat{Y}_s^T) \hat{Y}_s \hat{\Lambda}^{-\frac{1}{2}} = (Y_z \hat{Y}_s \hat{\Lambda}^{-\frac{1}{2}}) \hat{\Lambda} = \hat{V}\hat{\Lambda}.$

Für jede Ergänzung $V = [\hat{V}, \tilde{V}]$ mit $V^T V = I$ folgt $KV = V\Lambda$.

Kovarianz mehrerer Abtastungen

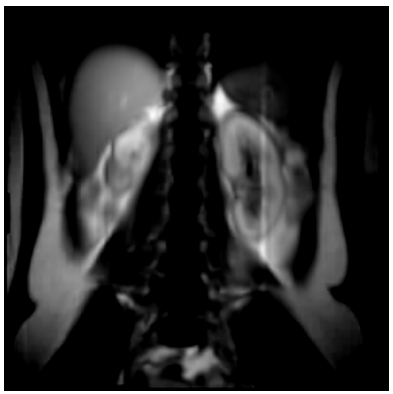
Beispiel: Für jede Zeit $t = 1, \ldots, T$ ist ein Bild $B(t)$ im Video,

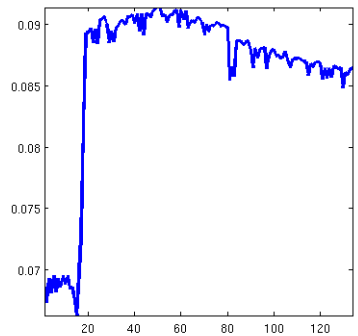
`http://math.uni-graz.at/invcon/medimage/respfilm1.mpg`

eine Matrix von Pixelwerten $B(t) = \{B_{i,j}(t)\}_{1 \le i,j \le N}$. Mit $m = T = 134$ und $n = N^2 = 400^2$ stellt man die Bilder als lange Vektoren dar:

$$\boldsymbol{y}_t^{\mathrm{T}} = \{\, B_{1,1}(t), \ldots, B_{N,1}(t), B_{1,2}(t), \ldots, B_{N,2}(t), \ldots, B_{1,N}(t), \ldots, B_{N,N}(t) \,\}$$

und führt die obigen Rechnungen mit $Y^{\mathrm{T}} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}$ durch. Dann sieht man links die este Zeile von Y_{s} in ein Bild umgewickelt,





während rechts die erste Spalte von V steht.

Mit der umgedrehten Datenstruktur $Y \leftarrow Y^{\mathrm{T}}$ gibt es viele Nullen in Λ. Mit der alternativen Zerlegung $Y_{\mathrm{z}}^{\mathrm{T}} Y_{\mathrm{z}} = \hat{Y}_{\mathrm{s}} \hat{\Lambda} \hat{Y}_{\mathrm{s}}^{\mathrm{T}}$ ist das Bild die erste Zeile von $\hat{Y}_s$, und der zeitliche Verlauf ist die erste Spalte von $\hat{V}$.

Korrelation

Die *Korrelation* zwischen $\boldsymbol{x} = \{x_i\}_{i=1}^n$ und $\boldsymbol{y} = \{y_i\}_{i=1}^n$ ist

$$\chi(\boldsymbol{x}, \boldsymbol{y}) = \frac{[\boldsymbol{x} - \mu(\boldsymbol{x})]^{\mathrm{T}}[\boldsymbol{y} - \mu(\boldsymbol{y})]}{\|\boldsymbol{x} - \mu(\boldsymbol{x})\|\|\boldsymbol{y} - \mu(\boldsymbol{y})\|} = \frac{\kappa(\boldsymbol{x}, \boldsymbol{y})}{\sigma(\boldsymbol{x})\sigma(\boldsymbol{y})}$$

Wegen der Cauchy-Schwarz Ungleichung $|\boldsymbol{a}^{\mathrm{T}}\boldsymbol{b}| \leq \|\boldsymbol{a}\|\|\boldsymbol{b}\|$,

$$|\chi(\boldsymbol{x}, \boldsymbol{y})| \leq 1.$$

Die *Korrelationsmatrix* ist

$$\mathrm{X} = \begin{bmatrix} \chi(\boldsymbol{x}, \boldsymbol{x}) & \chi(\boldsymbol{x}, \boldsymbol{y}) \\ \chi(\boldsymbol{y}, \boldsymbol{x}) & \chi(\boldsymbol{y}, \boldsymbol{y}) \end{bmatrix} = \begin{bmatrix} 1 & \chi(\boldsymbol{x}, \boldsymbol{y}) \\ \chi(\boldsymbol{x}, \boldsymbol{y}) & 1 \end{bmatrix}$$

Die Variablen sind *unkorreliert* wenn $\chi(\boldsymbol{x}, \boldsymbol{y}) = \kappa(\boldsymbol{x}, \boldsymbol{y}) = 0$ und

$$\mathrm{X} = I.$$

In diesem Fall gilt

$$K = \begin{bmatrix} \kappa(\boldsymbol{x}, \boldsymbol{x}) & \kappa(\boldsymbol{x}, \boldsymbol{y}) \\ \kappa(\boldsymbol{y}, \boldsymbol{x}) & \kappa(\boldsymbol{y}, \boldsymbol{y}) \end{bmatrix} = \begin{bmatrix} \sigma^2(\boldsymbol{x}) & 0 \\ 0 & \sigma^2(\boldsymbol{y}) \end{bmatrix}$$

und die Kovarianzmatrix ist diagonal.

Unabhängigkeit

Zufallsvariablen x und y sind unabhängig, wenn

$$P(x \in M \text{ und } y \in N) = P(x \in M) \cdot P(y \in N)$$

Beispiel: Zwei Münzen werden geworfen, und für

$$\text{1. Münze,} \quad x = \begin{cases} 1, & \text{Kopf} \\ 0, & \text{Zahl} \end{cases} \qquad \text{2. Münze,} \quad y = \begin{cases} 1, & \text{Kopf} \\ 0, & \text{Zahl} \end{cases}$$

Ein übliches Modell, das mit Erfahrung übereinstimmt, ist

$$P(x = m \text{ und } y = n) = P(x = m) \cdot P(y = n), \quad m, n \in \{0, 1\}$$

Bemerke: *Unabhängig* impliziert *Unkorrelliert*, aber nicht umgekehrt.

Beispiel: Punkte werden aus der Menge $\{(\pm 1, 0), (0, \pm 1)\}$ mit gleicher Häufigkeit ausgewählt, und $\boldsymbol{x}$ und $\boldsymbol{y}$ sind die x- bzw. y-Koordinaten der abgetasteten Punkte. Dann sind $\boldsymbol{x}$ und $\boldsymbol{y}$ unkorreliert:

$$\mu(\boldsymbol{x}) = \mu(\boldsymbol{y}) = 0$$

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \textstyle\sum_{i=1}^{n} (0)(\pm 1)/n = 0.$$

Sie sind aber nicht unabhängig:

$$0 = P(\boldsymbol{x} = 1 \text{ und } \boldsymbol{y} = 1) \neq P(\boldsymbol{x} = 1) \cdot P(\boldsymbol{y} = 1) = \tfrac{1}{4} \cdot \tfrac{1}{4} = \tfrac{1}{8}.$$

Mischungen von Abtastungen

Angenommen wären $\boldsymbol{z}_1$ und $\boldsymbol{z}_2$ ideale Abtastungen von 2 Quellen. Für das Folgende sind diese notwendigerweise unabhängig und nicht Gauß verteilt.

Dann werden tatsächlich abgetastet,

$$\left.\begin{array}{rcl} \boldsymbol{y}_1 & = & a_{11}\boldsymbol{z}_1 + a_{12}\boldsymbol{z}_2 \\ \boldsymbol{y}_2 & = & a_{21}\boldsymbol{z}_1 + a_{22}\boldsymbol{z}_2 \end{array}\right\} \text{ oder } Y = AZ \text{ mit } \left\{\begin{array}{rcl} Y^{\mathrm{T}} & = & \{\boldsymbol{y}_1, \boldsymbol{y}_2\} \\ Z^{\mathrm{T}} & = & \{\boldsymbol{z}_1, \boldsymbol{z}_2\} \end{array}\right.$$

Diese sind nicht mehr unabhängig: Beide haben einen Anteil von $\boldsymbol{z}_1$ und $\boldsymbol{z}_2$.

Nach dem Zentralen Grenzwertsatz sind $\boldsymbol{y}_1$ und $\boldsymbol{y}_2$ (d.h. Summen) mehr *Gauß* verteilt als $\boldsymbol{z}_1$ und $\boldsymbol{z}_2$!

Um diesen Effekt rückgängig zu machen, sucht man eine Transformation $W \approx A^{-1}$

$$\left.\begin{array}{rcl} \boldsymbol{x}_1 & = & w_{11}\boldsymbol{y}_1 + w_{12}\boldsymbol{y}_2 \\ \boldsymbol{x}_2 & = & w_{21}\boldsymbol{y}_1 + w_{22}\boldsymbol{y}_2 \end{array}\right\} \text{ oder } X = WY \text{ mit } \left\{\begin{array}{rcl} X^{\mathrm{T}} & = & \{\boldsymbol{x}_1, \boldsymbol{x}_2\} \\ Y^{\mathrm{T}} & = & \{\boldsymbol{y}_1, \boldsymbol{y}_2\} \end{array}\right.$$

wobei $\boldsymbol{x}_1$ und $\boldsymbol{x}_2$ am wenigsten Gauß verteilt wie möglich sind!

Mischungen von Abtastungen

Wenn $\boldsymbol{x}_1$ und $\boldsymbol{x}_2$ unabhängig sind, müssen sie auch unkorreliert sein. Daher werden $\boldsymbol{y}_1$ und $\boldsymbol{y}_2$ zuerst zentriert,

$$Y_z = Y - \overline{Y} = \begin{bmatrix} \boldsymbol{y}_1^T - \mu(\boldsymbol{y}_1) \\ \boldsymbol{y}_2^T - \mu(\boldsymbol{y}_2) \end{bmatrix} \in \mathbb{R}^{2\times 2}$$

und gesphärt durch $K = \frac{1}{n} Y_z Y_z^T$, $KV = V\Lambda$ und $VV^T = I$,

$$Y_s = \Lambda^{-\frac{1}{2}} V^T Y_z$$

damit sie unkorreliert sind $\frac{1}{n} Y_s Y_s^T = I$. Daher ist keine Drehung der Daten Y_s bevorzugt bezüglich Korrelation.

Jedoch kann eine Drehung U, $UU^T = I$, anschließend bestimmt werden,

$$X_z = UY_s = U\Lambda^{-\frac{1}{2}} V^T Y_z, \quad W = U\Lambda^{-\frac{1}{2}} V^T \approx A^{-1}$$

wobei die zentrierten Daten X_z am wenigsten Gauß verteilt wie möglich sind! Dann mit $X_z = UY_s = U\Lambda^{-\frac{1}{2}} V^T[Y - \overline{Y}]$ gilt

$$\begin{bmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \end{bmatrix} = X = X_z + U\Lambda^{-\frac{1}{2}} V^T \overline{Y} = U\Lambda^{-\frac{1}{2}} V^T Y = WY$$

und ähnlich für $m > 2$ Quellen.

Gaußianität

Welche Eigenschaften hat eine Gaußsche Verteilung, die zu vermeiden sind, um Gaußianität zu reduzieren?

Wenn eine Abtastung $\boldsymbol{n} = \{n_i\}_{i=1}^{n}$ nach der Verteilung $N(\mu, \sigma)$ Gauß verteilt ist, hat sie die *Momente*,

$$M_k(\boldsymbol{n}) = \frac{1}{n}\sum_{i=1}^{n} |n_i - \mu|^k \overset{n\to\infty}{\longrightarrow} \begin{cases} (k-1)!!\sigma^k, & k \text{ gerade} \\ \sqrt{\frac{2}{\pi}}(k-1)!!\sigma^k, & k \text{ ungerade} \end{cases}$$

wobei $k!! = k \cdot (k-2) \cdot (k-4) \cdots$ [2 oder 1].

Wölbung oder *Kurtosis* ist definiert für $\boldsymbol{x}$ (z.B. eine Zeile von $X_z = UY_s$) durch

$$W(\boldsymbol{x}) = M_4(\boldsymbol{x}) - 3M_2^2(\boldsymbol{x})$$

Für eine eher gleichmäßige Verteilung gilt $W(\boldsymbol{x}) < 0$. Für eine eher spitzige Verteilung gilt $W(\boldsymbol{x}) > 0$. Für ein Gauß verteiltes $\boldsymbol{n}$ gilt

$$W(\boldsymbol{n}) = 3\sigma^4 - 3\sigma^4 = 0.$$

Um Gaußianität zu minimieren, könnte man minimieren:

$J(\boldsymbol{u}) = -W^2(Y_s^T\boldsymbol{u})$ für jede Spalte in $U^T = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m\}$.

Hauptkomponentenanalyse (PCA) und Unabhängigkeitsanalyse (ICA)

(PCA) Seien die Daten $Y^{\mathrm{T}} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}$ zentriert,

$$Y_{\mathrm{z}} = [Y - \overline{Y}] \in \mathbb{R}^{m \times n}, \quad \overline{Y}^{\mathrm{T}} = \{\mu(\boldsymbol{y}_1), \ldots, \mu(\boldsymbol{y}_m)\}$$

und gesphärt durch $K = \frac{1}{n} Y_{\mathrm{z}} Y_{\mathrm{z}}^{\mathrm{T}}$, $KV = V\Lambda$ und $VV^{\mathrm{T}} = I$,

$$Y_{\mathrm{s}} = \Lambda^{-\frac{1}{2}} V^{\mathrm{T}} Y_{\mathrm{z}}$$

Sei $\Lambda = \mathrm{diag}\{\lambda_1, \ldots, \lambda_m\}$ mit $\lambda_1 \geq \cdots \geq \lambda_m$. Dann sind die späteren Zeilen in Y_{s} *schwächer* vertreten in den Daten Y als die früheren. Mit $P \in \mathbb{R}^{r \times m}$, $r < m$, $P_{ij} = \delta_{ij}$ ($\delta_{ij} = 1, i = j; \delta_{ij} = 0, i \neq j$) werden die Daten Y auf die *r stärksten* Hauptkomponenten so projiziert:

$$Y \approx Y_P = \overline{Y} + V\Lambda^{\frac{1}{2}} P^{\mathrm{T}} P Y_{\mathrm{s}} = \overline{Y} + \frac{1}{n} Y_{\mathrm{z}} (PY_{\mathrm{s}})^{\mathrm{T}} (PY_{\mathrm{s}})$$

(ICA) Sei die Drehungsmatrix U so bestimmt,

$$X_{\mathrm{z}} = UY_{\mathrm{s}} = U\Lambda^{-\frac{1}{2}} V^{\mathrm{T}} Y_{\mathrm{z}}$$

dass die Zeilen der zentrierten Daten X_{z} optimal unabhängig sind. Mit $Q \in \mathbb{R}^{r \times m}$, $r < m$, und $Q_{i,j} = \delta_{q_i,j}$, werden die Daten Y auf die *r* unabhängigen Komponenten $\{q_1, \ldots, q_r\}$ so projiziert:

$$Y \approx Y_Q = \overline{Y} + V\Lambda^{\frac{1}{2}} U^{\mathrm{T}} Q^{\mathrm{T}} Q X_{\mathrm{z}} = \overline{Y} + \frac{1}{n} Y_{\mathrm{z}} (QX_{\mathrm{z}})^{\mathrm{T}} (QX_{\mathrm{z}})$$

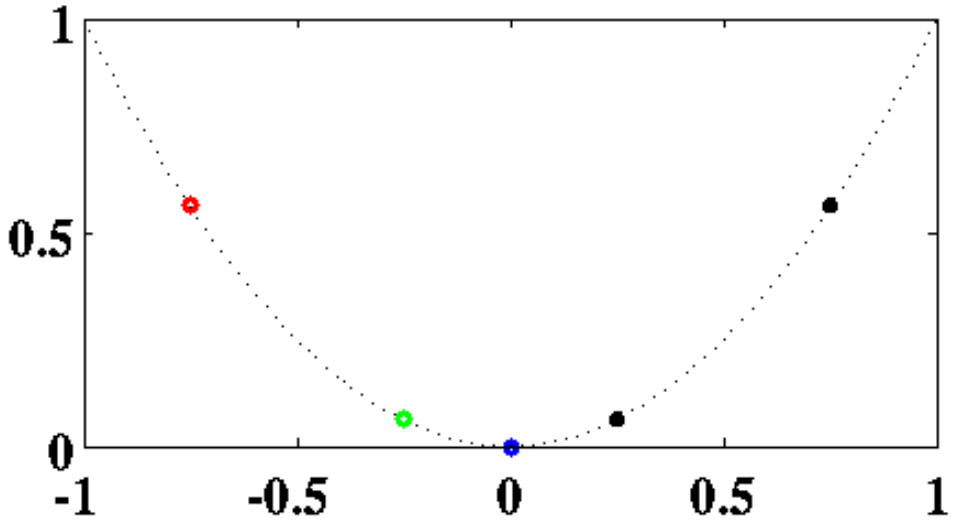
Optimierung

Zuerst wird überlegt, wie man eine Funktion $f(x)$, $x \in \mathbb{R}^1$, minimiert, wenn man nur $f(x)$ (und keine Ableitungen) in einer *black-box* Weise auswerten kann.

Beispiel: Man beginnt mit 2 Startpunkten, z.B.

$$x_1 = \frac{1}{4}, \quad x_2 = \frac{3}{4}$$

und wertet aus, $f(x_1) < f(x_2)$. (Kurve unbekannt)



Die nächsten vernünftigen Proben sind

$x_1 - 2(x_2 - x_1) = -\frac{3}{4}$, $\quad x_1 - (x_2 - x_1) = -\frac{1}{4}$, $\quad x_1 - \frac{1}{2}(x_2 - x_1) = 0$

Es wird ersetzt $x_2 \leftarrow x_1 - \frac{1}{2}(x_2 - x_1)$, dann neu gestartet, usw.

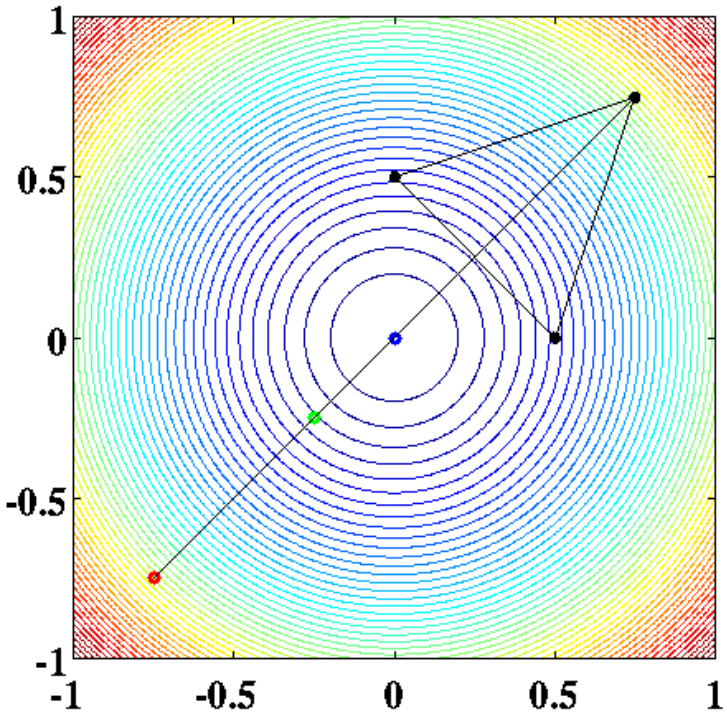
Nelder-Mead Verfahren

Dieses Verfahren, nach Nelder und Mead, heisst auch *Simplex Verfahren*, besonders in höheren Dimensionen.

Beispiel: In $\mathbb{R}^2$ beginnt man mit 3 Startpunkten, z.B.

$$\boldsymbol{x}_1 = (\tfrac{3}{4}, \tfrac{3}{4}), \quad \boldsymbol{x}_2 = (0, \tfrac{1}{2}), \quad \boldsymbol{x}_3 = (\tfrac{1}{2}, 0)$$

und wertet aus, $f(\boldsymbol{x}_1) > f(\boldsymbol{x}_2) = f(\boldsymbol{x}_3)$. (Konturen unbekannt)



Mit dem Mittelpunkt

$$\boldsymbol{x}_\mathrm{m} = \frac{\boldsymbol{x}_2 + \boldsymbol{x}_3}{2}$$

sind die nächsten vernünftigen Proben:

$$\boldsymbol{x}_\mathrm{m} - \tfrac{1}{2}(\boldsymbol{x}_1 - \boldsymbol{x}_\mathrm{m})$$

$$\boldsymbol{x}_\mathrm{m} - (\boldsymbol{x}_1 - \boldsymbol{x}_\mathrm{m})$$

und

$$\boldsymbol{x}_\mathrm{m} - 2(\boldsymbol{x}_1 - \boldsymbol{x}_\mathrm{m})$$

Es wird ersetzt $\boldsymbol{x}_1 \leftarrow \boldsymbol{x}_\mathrm{m} - \frac{1}{2}(\boldsymbol{x}_1 - \boldsymbol{x}_\mathrm{m})$, dann neu gestartet, usw.

Nelder-Mead Verfahren

MATLAB Code für diese Grafik:

```
n = 501;
x = kron(linspace(-1,1,n),ones(n,1));
y = x';
z = x.^2 + y.^2;
contour(x,y,z,50);
hold on;
plot(+0.75,+0.75,'k*', ...
     +0.50,+0.00,'k*', ...
     +0.00,+0.50,'k*', ...
     +0.00,+0.00,'bo', ...
     -0.25,-0.25,'go', ...
     -0.75,-0.75,'ro');
plot([0,0.5,0.75,0],[0.5,0,0.75,0.5],'k',...
     [-0.75,0.75],[-0.75,0.75],'k');
hold off;
```

fminsearch

MATLAB Code zur Simplex Minimierung von $f(x) = x^2 + y^2$ über `MyMin.m` mit der eingebauten Funktion `fminsearch`:

```
function MyMin
opts = optimset('MaxIter',300,'MaxFunEvals',500);
x0 = [1,1];
[x,fval] = fminsearch(@MyFun,x0,opts)

function f = MyFun(x)
f = sum(x.^2);
```

Im Befehlsfenster,

```
>> MyMin
x =
   1.0e-04 *
     -0.2102  0.2548
fval =
   3.3037e-05
```

Optimierung der Wölbung mit Nelder-Mead

Gegeben sind Signale $\boldsymbol{y}_1$ und $\boldsymbol{y}_2$, Mischungen von unabhängigen (nicht Gaußschen) $\boldsymbol{z}_1$ und $\boldsymbol{z}_2$.

MATLAB Code-Stück zur Erzeugung von

$$Z^{\mathrm{T}} = \{\boldsymbol{z}_1, \boldsymbol{z}_2\} \text{ und } Y^{\mathrm{T}} = \{\boldsymbol{y}_1, \boldsymbol{y}_2\} \text{ mit } Y = AZ$$

für die Mischungsmatrix $A = [4, 3; 2, 1]$:

```
global n Ys                    (Siehe global unten in MyFun)
n = 101;
t = linspace(0,10,n);
A = [4,3;2,1];
Z = [rand(1,n);t/2-floor(t/2)];
Y = A*Z;
```

Die Daten $\boldsymbol{y}_1$ und $\boldsymbol{y}_2$ werden zentriert,

$$Y_{\mathrm{z}} = [Y - \overline{Y}] \in \mathbb{R}^{2\times n} \quad \overline{Y} = \{\mu(\boldsymbol{y}_1), \mu(\boldsymbol{y}_2)\}^{\mathrm{T}}$$

und gesphärt durch $K = \frac{1}{n} Y_{\mathrm{z}} Y_{\mathrm{z}}^{\mathrm{T}}$, $KV = V\Lambda$ und $VV^{\mathrm{T}} = I$,

$$Y_{\mathrm{s}} = \Lambda^{-\frac{1}{2}} V^{\mathrm{T}} Y_{\mathrm{z}}$$

Dann mit $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} = 1$ folgt $M_2(Y_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u}) = \frac{1}{n}\boldsymbol{u}^{\mathrm{T}} Y_{\mathrm{s}} Y_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u} = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} = 1$.

Optimierung der Wölbung mit Nelder-Mead

MATLAB Code-Stück, um die Daten zu zentrieren und sphären:

```
Yz(1,:)  = Y(1,:)  - mean(Y(1,:));
Yz(2,:)  = Y(2,:)  - mean(Y(2,:));
K = Yz*Yz'/n; [V,D] = eig(K);
Ys = diag(1./sqrt(diag(D)))*V'*Yz;
```

Der erste Vektor $\boldsymbol{u}_1$ in $U^{\mathrm{T}} = \{\boldsymbol{u}_1, \boldsymbol{u}_2\}$ soll bestimmt werden, um

$$J(\boldsymbol{u}_1) = -W^2(Y_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u}_1) = -[\textstyle\sum_{i=1}^{n}(\boldsymbol{e}_i Y_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u}_1)^4/n - 3]^2$$

zu minimieren ▸. Da $UU^{\mathrm{T}} = I$ gelten muss, folgen

$$\boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{u}_1 = \boldsymbol{u}_2^{\mathrm{T}}\boldsymbol{u}_2 = 1 \quad \text{und} \quad \boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{u}_2 = 0$$

Daher mit $\boldsymbol{u}_1 = \langle u_1, u_2\rangle^{\mathrm{T}}$ gegeben, passt $\boldsymbol{u}_2 = \langle -u_2, u_1\rangle^{\mathrm{T}}$.

MATLAB Code-Stück zur Bestimmung von U und daher X_{z}:

```
u0 = [1;0];
opts = optimset('MaxIter',300,'MaxFunEvals',500);
u = fminsearch(@MyFun,u0,opts);
u = u/norm(u);
U = [u(1),u(2);-u(2),u(1)]; Xz = U*Ys;
```

Optimierung der Wölbung mit Nelder-Mead

MATLAB Code-Stück zur Auswertung der Wölbung und Zielfunktion für den obigen Aufruf über *fminsearch*:

```
function f = MyFun(u)
global n Ys                                   (Siehe global oben)
u1 = u/norm(u);       % M2 = mean((u1'*Ys).^2)->1
M4 = mean((u1'*Ys).^4);
f = -(M4 - 3)^2;
```

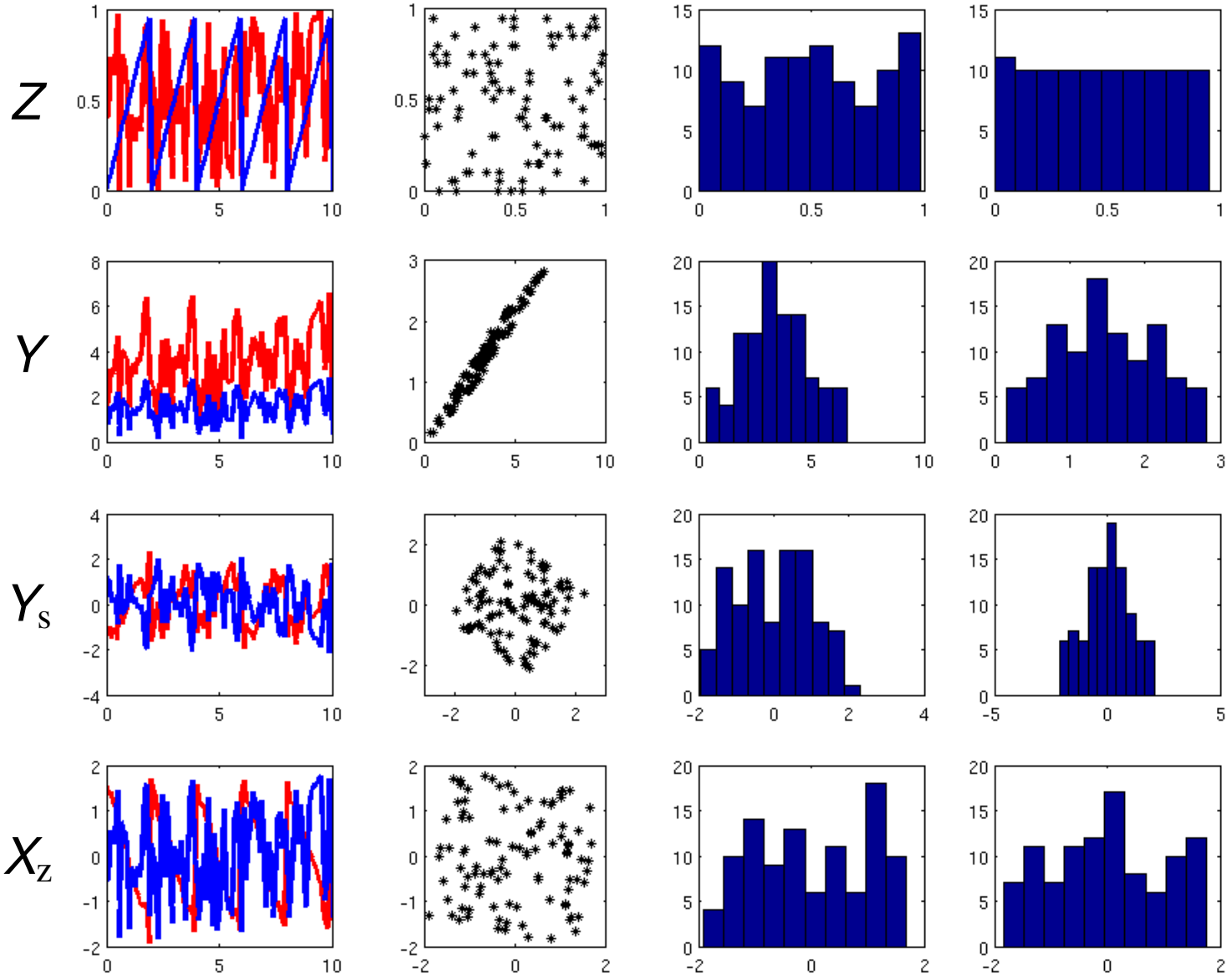
Diese Funktion kann auch aufgerufen werden, um die Landschaft der Zielfunktion grafisch darzustellen:

```
km = 20; th = zeros(km,1); J = zeros(km,1);
for k=1:km
   th(k) = 2*pi*(k-1)/km;
   u = [cos(th(k));sin(th(k))];          % ||u||=1
   J(k) = MyFun(u);
end
plot(th,J)
```

Hier wird mit $\boldsymbol{u}(\theta) = \{\cos(\theta); \sin(\theta)\}$, $\theta \in [0, 2\pi]$, parameterisiert.

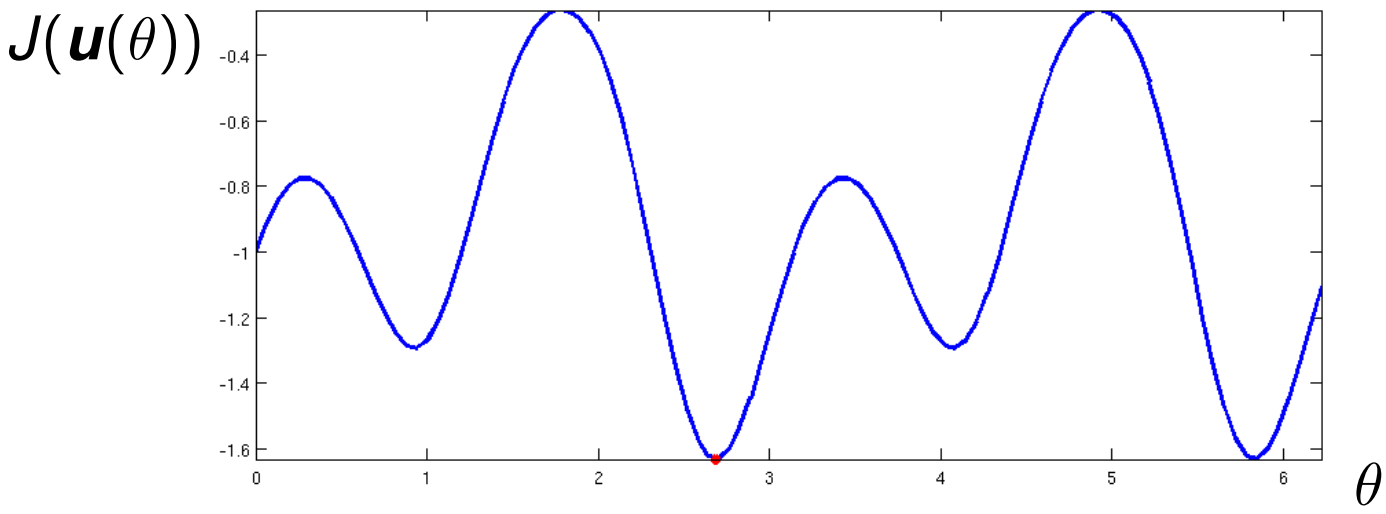
Optimierung der Wölbung mit Nelder-Mead

Ergebnis der Optimierung: Quellen erfolgreich getrennt!



Optimierung der Wölbung mit Nelder-Mead

Die Zielfunktion für das obige Beispiel mit $m = 2$,



Bemerkungen:

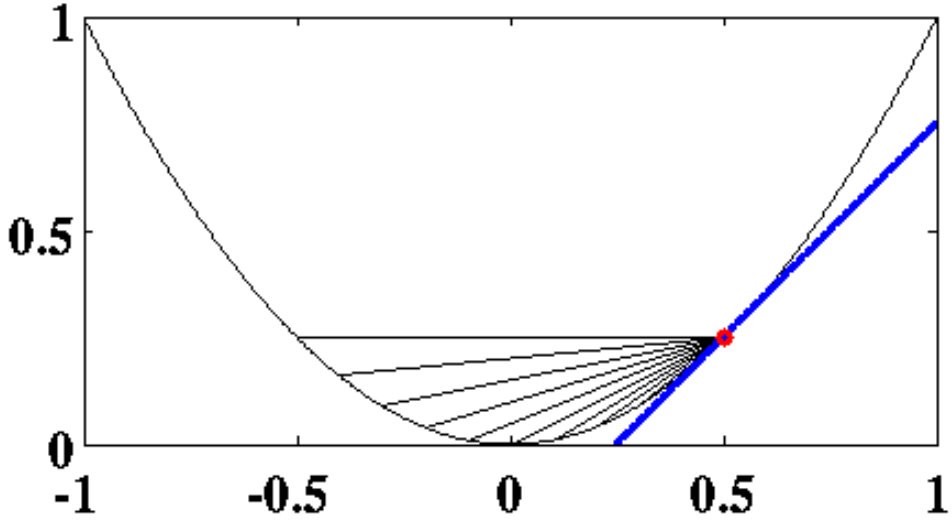
- Die Zielfunktion $J(\boldsymbol{u}(\theta)) = -W^2(Y_s^T \boldsymbol{u}(\theta))$ ist π-periodisch,
- und π-Differenzen in Drehungen werden gleich beurteilt.
- Die Zielfunktion hat lokale Minima,
- und breite und flache Gebiete sind üblich.
- Obwohl das Nelder-Mead Verfahren oft geeignet für globale Optimierung ist,
- kann eine globale Suche mit roher Gewalt notwendig sein.

Abstiegsverfahren

Nun wird überlegt, wie man eine Funktion $f(x)$, $x \in \mathbb{R}^1$, minimiert, wenn man die Funktion $f(x)$ und ihre Ableitung $f'(x)$ auswerten kann.

Die Ableitung $f'(x)$ an der Stelle x ist der Limes der Steigungen von Sekantenstrecken:

$$D_x f(x) = f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{(x+h) - h} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$



Sei $f(x) = x^2$ mit $f'(x) = 2x$. An der Stelle $x = \frac{1}{2}$ gilt $f'(\frac{1}{2}) = 1$. Da $f'(\frac{1}{2}) > 0$ gilt, erfüllen die nächsten vernünftigen Proben x zur Minimierung von $f(x)$ die Ungleichung $x < \frac{1}{2}$.

Abstiegsverfahren

Das Abstiegsverfahren ist

$$\text{(neues } x\text{)} \quad x \leftarrow x - \alpha f'(x) \quad \text{(altes } x\text{)}$$

wobei das neue x das alte x ersetzt, und α wird ausgewählt, sodass $f(x)$ reduziert wird.

Wie beim Simplex Verfahren sind vernünftige Proben für α gegeben durch

$$\alpha \in \{0, \tfrac{1}{2}, 1, 2\}$$

und das α wird gewählt, wo f am kleinsten ist.

MATLAB Code zur Minimierung von $f(x) = x^2$:

```
f=@(x) x.^2; fp=@(x) 2*x; a=[0,1,2,4]/2;
imax=100; tol=1.0e-3; x=0.5;
for i=1:imax
   xa=x; fpx=fp(x); fv=f(x-a*fpx);
   j=find(fv == min(fv),1); x=x-a(j)*fpx;
   if (abs(x-xa) <= tol*abs(x)); break; end;
end
```

Abstiegsverfahren für Systeme

In $\mathbb{R}^2$ (oder höher) verwendet man *Richtungsableitungen* für das Abstiegsverfahren.

Die Richtungsableitung $D_{\boldsymbol{u}} f(\boldsymbol{x})$ an der Stelle $\boldsymbol{x} = (x, y)$ in der Richtung $\boldsymbol{u} = (u, v)$ ($\|\boldsymbol{u}\| = 1$) ist der Limes der Steigungen von Sekantenstrecken in einer senkrechten Ebene durch $\boldsymbol{x}$ und $\boldsymbol{x} + \boldsymbol{u}$:

$$D_{\boldsymbol{u}} f(\boldsymbol{x}) = \lim_{h \to 0} \frac{f(\boldsymbol{x} + h\boldsymbol{u}) - f(\boldsymbol{x})}{h} = \cdots = D_x f(\boldsymbol{x}) u + D_y f(\boldsymbol{x}) v$$

Für $f(x, y) = x^2 + y^2$, $\boldsymbol{x}_0 = (1, 1)$ und $\boldsymbol{u} = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ gelten

$$D_x f(x) = 2x, \quad D_y f(x) = 2y$$

und

$$D_{\boldsymbol{u}} f(\boldsymbol{x}_0) = D_x f(\boldsymbol{x}_0) u + D_y f(\boldsymbol{x}_0) v = 2\tfrac{1}{\sqrt{2}} + 2\tfrac{1}{\sqrt{2}} = 2\sqrt{2} > 0.$$

Daher läuft f bergauf an der Stelle $\boldsymbol{x}_0$ in der Richtung $\boldsymbol{u}$.

Die Richtung des steilsten Anstiegs ist

$$\boldsymbol{u}^*(\boldsymbol{x}) = \langle D_x f, D_y f \rangle^{\mathrm{T}} / \| \langle D_x f, D_y f \rangle \|$$

und das Abstiegsverfahren ist,

$$\text{(neues } \boldsymbol{x}) \quad \boldsymbol{x} \leftarrow \boldsymbol{x} - \alpha \boldsymbol{u}^*(\boldsymbol{x}) \quad \text{(altes } \boldsymbol{x})$$

für das minimierende $\alpha \in \{0, \frac{1}{2}, 1, 2\}$.

Abstiegsverfahren für Systeme

Optimierung der Wölbung? Die zu minimierende Funktion ist

$$J(\boldsymbol{u}) = -W^2(Y_s^T\boldsymbol{u}) = -[M_4(Y_s^T\boldsymbol{u}) - 3M_2^2(Y_s^T\boldsymbol{u})]^2$$
$$= -[M_4(Y_s^T\boldsymbol{u}) - 3]^2$$

Der Gradient dieser Funktion ist mit der Kettenregel

$$D_{\boldsymbol{u}}J(\boldsymbol{u}) = -2[M_4(Y_s^T\boldsymbol{u}) - 3]D_{\boldsymbol{u}}M_4(Y_s^T\boldsymbol{u})$$

wobei die Gradienten der Momente sind

$$D_{\boldsymbol{u}}M_k(Y_s^T\boldsymbol{u}) = \frac{k}{n}\sum_{i=1}^{n}(\boldsymbol{e}_i^T Y_s^T\boldsymbol{u})^{k-1}(Y_s\boldsymbol{e}_i)$$

Die Richtung des steilsten Anstiegs ist

$$\boldsymbol{u}^*(\boldsymbol{x}) = D_{\boldsymbol{u}}J(\boldsymbol{u})/\|D_{\boldsymbol{u}}J(\boldsymbol{u})\|$$

und das Abstiegsverfahren ist,

$$\text{(neues } \boldsymbol{u}) \quad \boldsymbol{u} \leftarrow \boldsymbol{u} - \alpha\boldsymbol{u}^*(\boldsymbol{u}) \quad \text{(altes } \boldsymbol{u})$$

für das minimierende $\alpha \in \{0, \frac{1}{2}, 1, 2\}$, wobei mit

$$\boldsymbol{u} \leftarrow \boldsymbol{u}/\|\boldsymbol{u}\|$$

normalisiert wird. Auch wenn $\boldsymbol{u}^\star$ schon für eine andere Komponente bestimmt worden ist, wird $\boldsymbol{u}$ so projiziert

$$\boldsymbol{u} \leftarrow \boldsymbol{u} - \boldsymbol{u}^\star(\boldsymbol{u}^T\boldsymbol{u}^\star).$$

Optimierung der Wölbung mit Roher Gewalt

MATLAB Code zur Abschätzung der Drehung U mit roher Gewalt:

```
im = 1000; U = []; I = eye(m);
Ms = 3;                          % M2 = mean(xz.^2)->1
for k=1:m
   Jm = 0;
   for i=1:im
      u = randn(m,1);
      if (k > 1) u = (I - U'*U)*u; end
      u = u/norm(u);
      xz = u'*Ys; M4 = mean(xz.^4);      % u = 0?
      J = -(M4 - Ms)^2;
      if (J < Jm) Jm = J; um = u; end
   end
   U = [U;um'];
end
Us = U; Xz = Us*Ys;
```

$Y_s \in \mathbb{R}^{m \times n}$

$U^T = \{u^\star, \ldots\} \Rightarrow u^T u^\star = 0$

$u^T u = 1$

Optimierung der Wölbung mit Abstiegsverfahren

MATLAB Code um U mit dem Abstiegsverfahren endgültig zu bestimmen, wobei mit der Abschätzung U_s vom letzten Code gestartet wird:

```
tol = 1.0e-4; itmax = 1000; U = []; lm = 4;
Ms = 3;                          % M2 = mean(xz.^2)->1
for k=1:m
   u = Us(k,:)';
   if (k > 1) u = (I - U'*U)*u; end
   u = u/norm(u);                                % u = 0?
   for it=1:itmax
      ua = u;
      xz = u'*Ys;
      M4 = mean(xz.^4);
      DM4 = 4*mean(spdiags(xz(:).^3,0,n,n)*Ys');
      DJ = -2*(M4 - Ms)*DM4';
      J = -(M4 - Ms)^2;
```

Optimierung der Wölbung mit Abstiegsverfahren

```
      for l=1:lm
        al = 2*l/lm; ut = u - al*DJ;
        if (k > 1) ut = (I - U'*U)*u; end
        ut = ut/norm(ut);
        xt = ut'*Ys;
        M4 = mean(xt.^4);
        J = [J,-(M4 - Ms)^2];
      end
      l = find(J == min(J(:)),1)-1; al = 2*l/lm;
      u = u - al*DJ;
      if (k > 1) u = (I - U'*U)*u; end
      u = u/norm(u); du = (I - ua*ua')*u;   % u = 0?
      if (max(abs(du)) <= tol) break; end
    end
    U = [U;u'];
  end
  Us = U; Xz = Us*Ys;
```

Newton Verfahren

Nun wird überlegt, wie man eine Funktion $f(x)$, $x \in \mathbb{R}^1$, minimiert, wenn man die Funktion $f(x)$ und ihre Ableitungen $f'(x)$ und $f''(x)$ auswerten kann.

Die Strategie: Eine Nullstelle $f'(x) = 0$ wird durch eine Folge von linearen Approximationen bestimmt:

Für ein gegebenes x_0 ist

$$y - f'(x_0) = f''(x_0)(x - x_0)$$

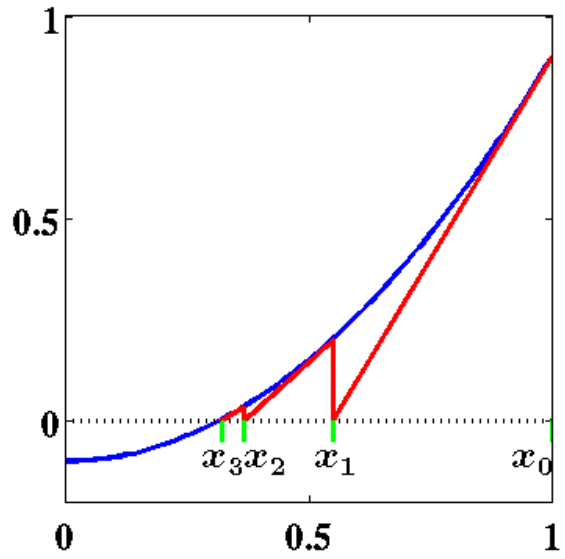
die zur Kurve $f'(x)$ tangente Gerade an der Stelle $(x_0, f'(x_0))$.

Die Nullstelle dieser Gerade erfüllt

$$0 - f'(x_0) = f''(x_0)(x - x_0)$$

oder

$$x_1 \leftarrow x = x_0 - f'(x_0)/f''(x_0)$$



Newton Verfahren

Das Newton Verfahren ist

$$\text{(neues } x\text{)} \quad x \leftarrow x - \alpha f'(x)/f''(x) \quad \text{(altes } x\text{)}$$

wobei das neue x das alte x ersetzt, und α wird ausgewählt, sodass $f(x)$ reduziert wird.

Wie beim Simplex Verfahren sind vernünftige Proben für α gegeben durch

$$\alpha \in \{0, \tfrac{1}{2}, 1, 2\}$$

und das α wird gewählt, wo f am kleinsten ist.

MATLAB Code zur Minimierung von $f(x) = x^2$:

```
f=@(x) x.^2; fp=@(x) 2*x; fpp=@(x) 2;
imax=100; tol=1.0e-3; x=0.5; a=[0,1,2,4]/2;
for i=1:imax
   xa=x; fpx=fp(x); fppx=fpp(x); fv=f(x-a*fpx/fppx);
   j=find(fv == min(fv),1); x=x-a(j)*fpx/fppx;
   if (abs(x-xa) <= tol*abs(x)); break; end;
end
```

Newton Verfahren für Systeme

In $\mathbb{R}^2$ (oder höher) verwendet man den Gradienten,

$$Df(\boldsymbol{x}) = \langle D_x f(\boldsymbol{x}), D_y f(\boldsymbol{x}) \rangle^{\mathrm{T}}$$

und die Hesse Matrix von f,

$$D^2 f(\boldsymbol{x}) = \begin{bmatrix} D_x^2 f(\boldsymbol{x}) & D_x D_y f(\boldsymbol{x}) \\ D_y D_x f(\boldsymbol{x}) & D_y^2 f(\boldsymbol{x}) \end{bmatrix}$$

Das Newton Verfahren ist

$$D^2 f(\boldsymbol{x})\boldsymbol{d} = -Df(\boldsymbol{x})$$

und

$$(\text{neues } \boldsymbol{x}) \quad \boldsymbol{x} \leftarrow \boldsymbol{x} + \alpha \boldsymbol{d} \quad (\text{altes } \boldsymbol{x})$$

wobei das neue $\boldsymbol{x}$ das alte $\boldsymbol{x}$ ersetzt, und $\alpha \in \{0, \frac{1}{2}, 1, 2\}$ wird ausgewählt, sodass $f(\boldsymbol{x})$ reduziert wird. Wenn $D^2 f(\boldsymbol{x})$ positiv definit ist, gilt

$$0 < \alpha \boldsymbol{d}^{\mathrm{T}} D^2 f(\boldsymbol{x}) \boldsymbol{d} = -\alpha \boldsymbol{d}^{\mathrm{T}} Df(\boldsymbol{x}) = -(\boldsymbol{x}_{\mathrm{neu}} - \boldsymbol{x}_{\mathrm{alt}})^{\mathrm{T}} Df(\boldsymbol{x})$$

und so $(\boldsymbol{x}_{\mathrm{neu}} - \boldsymbol{x}_{\mathrm{alt}})$ hat eine Richtung ähnlich zu $-Df(\boldsymbol{x})$ mit dem steilsten Abstieg. Daher wählt man $\alpha > 0$ aus.

Optimierung von Wölbung? Die Hesse Matrix für $J(\boldsymbol{u})$ ist:

Newton Verfahren für Systeme

$$D^2_{\boldsymbol{u}} J(\boldsymbol{u}) = -2\boldsymbol{M}\boldsymbol{M}^{\mathrm{T}} + \textit{vernachlässigbar}$$

mit $$\boldsymbol{M} = D_{\boldsymbol{u}} M_4(Y_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u})$$

wobei $D_{\boldsymbol{u}} M_k(Y_{\mathrm{s}}^{\mathrm{T}}\boldsymbol{u})$ früher für $D_{\boldsymbol{u}} J(\boldsymbol{u})$ berechnet worden sind.

Die Approximation $D^2_{\boldsymbol{u}} J(\boldsymbol{u}) \approx -2\boldsymbol{M}\boldsymbol{M}^{\mathrm{T}}$ ist negativ *semi*-definit,

$$\boldsymbol{d}^{\mathrm{T}} D^2_{\boldsymbol{u}} J(\boldsymbol{u})\boldsymbol{d} \approx -2\boldsymbol{d}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{M}^{\mathrm{T}}\boldsymbol{d} = -2(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{d})^2 \leq 0$$

und daher wird eine Abstiegsrichtung durch $\alpha < 0$ unten gesichert,

$$\alpha\boldsymbol{d}^{\mathrm{T}} D_{\boldsymbol{u}} J(\boldsymbol{u})\boldsymbol{d} \approx -\alpha\boldsymbol{d}^{\mathrm{T}} D_{\boldsymbol{u}} J(\boldsymbol{u}) = -(\boldsymbol{u}_{\mathrm{neu}} - \boldsymbol{u}_{\mathrm{alt}})^{\mathrm{T}} D_{\boldsymbol{u}} J(\boldsymbol{u}) > 0!$$

Für ein kleines $\epsilon > 0$ ist das approximierte Newton Verfahren,

$$(-2\boldsymbol{M}\boldsymbol{M}^{\mathrm{T}} - \epsilon I)\boldsymbol{d} = -D_{\boldsymbol{u}} J(\boldsymbol{u})$$

und

$$(\text{neues } \boldsymbol{u}) \quad \boldsymbol{u} \leftarrow \boldsymbol{u} + \alpha\boldsymbol{d} \quad (\text{altes } \boldsymbol{u})$$

für das minimierende $\alpha \in \{0, -\frac{1}{2}, -1, -2\}$, wobei mit

$$\boldsymbol{u} \leftarrow \boldsymbol{u}/\|\boldsymbol{u}\|$$

normalisiert wird. Auch wenn $\boldsymbol{u}^\star$ schon für eine andere Komponente bestimmt worden ist, wird $\boldsymbol{u}$ so projiziert

$$\boldsymbol{u} \leftarrow \boldsymbol{u} - \boldsymbol{u}^\star(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}^\star).$$

Optimierung der Wölbung mit Newton Verfahren

MATLAB Code um U mit dem Newton Verfahren endgültig zu bestimmen, wobei mit der Abschätzung U_s vom letzten Code gestartet wird:

```
tol = 1.0e-7; itmax = 100; U = []; lm = 4;
I = eye(m); Ms = 3;          % M2 = mean(xz.^2)->1
for k=1:m
   u = Us(k,:)';
   if (k > 1) u = (I - U'*U)*u; end
   u = u/norm(u);                          % u = 0?
   for it=1:itmax
      ua = u;
      xz = u'*Ys;
      M4 = mean(xz.^4);
      DM4 = 4*mean(spdiags(xz(:).^3,0,n,n)*Ys');
      DJ = -2*(M4 - Ms)*DM4';
      M = DM4; ep = 1.0e-3*max(abs(M));
      d = (2*M*M'+ep*I)\DJ;
```

Optimierung der Wölbung mit Newton Verfahren

```
        J = -(M4 - Ms)^2;
        for l=1:lm
          al = -2*l/lm; ut = u + al*d;
          if (k > 1) ut = (I - U'*U)*u; end
          ut = ut/norm(ut);
          xt = ut'*Ys;
          M4 = mean(xt.^4);
          J = [J,-(M4 - Ms)^2];
        end
        l = find(J == min(J(:)),1)-1; al = -2*l/lm;
        u = u + al*d;
        if (k > 1) u = (I - U'*U)*u; end    % u = 0?
        u = u/norm(u); du = (I - ua*ua')*u;
        if (max(abs(du)) <= tol) break; end
      end
      U = [U;u'];
    end
    Xz = U*Ys;
```

Fortgeschrittene Themen

Eine alternative Zielfunktion wird mit dem folgenden Beispiel motiviert.

Basierend auf dem obigen Beispiel seien $\boldsymbol{y}_1$ und $\boldsymbol{y}_2$ in

$$Y = \{\boldsymbol{y}_1, \boldsymbol{y}_2\}^{\mathrm{T}} \in \mathbb{R}^{2\times n}$$

die ersten bwz. zweiten Koordinaten der abgetasteten Punkte, die aus der Menge $\{(\pm 1, 0), (0, \pm 1)\}$ mit gleicher Häufigkeit ausgewählt werden, aber es gibt *genau zwei Ausreißer* :

$$(\boldsymbol{y}_1)_1 = \alpha, \quad (\boldsymbol{y}_2)_1 = 0$$
$$(\boldsymbol{y}_1)_2 = -\alpha, \quad (\boldsymbol{y}_2)_2 = 0$$

Sonst gelten $(\boldsymbol{y}_1)_3 = (\boldsymbol{y}_1)_4 = 0$, $(\boldsymbol{y}_2)_3 = -(\boldsymbol{y}_2)_4 = 1$,

$$\overline{Y}^{\mathrm{T}} = \{\mu(\boldsymbol{y}_1), \mu(\boldsymbol{y}_2)\} = 0$$

und

$$\sigma^2(\boldsymbol{y}_1) = \frac{1}{n}\left[\underbrace{2\alpha^2}_{i=1,2} + \underbrace{0}_{i=3,4} + \sum_{i=5}^{n}(\boldsymbol{y}_1)_i^2\right]$$
$$= \frac{1}{n}\left[2\alpha^2 + \frac{1}{2}(n-4)\right] = \frac{1}{2}\left[1 + \frac{4}{n}(\alpha^2 - 1)\right]$$

Robuste Zielfunktion

Ähnlich,

$$\sigma^2(\boldsymbol{y}_2) = \frac{1}{n}\left[\underbrace{0}_{i=1,2} + \underbrace{2}_{i=3,4} + \sum_{i=5}^{n}(\boldsymbol{y}_2)_i^2\right] = \frac{1}{n}\left[2 + \frac{1}{2}(n-4)\right] = \frac{1}{2}$$

Mit $Y_z = [Y - \overline{Y}] = Y$,

$$K = \tfrac{1}{n} Y_z Y_z^T = \tfrac{1}{n} YY^T = \text{diag}\{\sigma^2(\boldsymbol{y}_1), \sigma^2(\boldsymbol{y}_2)\} := \Lambda$$

und $V = I$ gilt $KV = V\Lambda$, und die Daten werden so gesphärt,

$$Y_s = \Lambda^{-\frac{1}{2}} V Y_z = \left\{\frac{\boldsymbol{y}_1}{\sigma(\boldsymbol{y}_1)}, \frac{\boldsymbol{y}_2}{\sigma(\boldsymbol{y}_2)}\right\}^T, \quad \frac{1}{n} Y_s Y_s^T = I$$

Die Zeilen von Y_s sind unkorreliert aber nicht unabhängig ▸.

Ohne Ausreißer sind die Daten unabhängig mit der Drehung

$$U = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}, \quad \theta = \frac{\pi}{4}, \quad \boldsymbol{u}^\star = \{1,1\}/\sqrt{2}$$

Eine Zeile $\boldsymbol{u} \approx \boldsymbol{u}^\star$ von U wird gesucht, wobei $\boldsymbol{x} = Y_s^T \boldsymbol{u}$ möglichst nicht Gauß verteilt ist. Für die Wölbung wird berechnet,

$$M_2(Y_s^T \boldsymbol{u}) = \frac{1}{n} \boldsymbol{u}^T Y_s Y_s^T \boldsymbol{u} = \boldsymbol{u}^T \boldsymbol{u} = 1$$

Robuste Zielfunktion

und

$$M_4(Y_s^T \boldsymbol{u}) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{e}_i^T Y_s^T \boldsymbol{u})^4 = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{(\boldsymbol{y}_1)_i}{\sigma(\boldsymbol{y}_1)}u_1 + \frac{(\boldsymbol{y}_2)_i}{\sigma(\boldsymbol{y}_2)}u_2\right]^4$$

$$= \frac{1}{n}\left\{\sum_{i=1}^{2}\left[\frac{(\boldsymbol{y}_1)_i}{\sigma(\boldsymbol{y}_1)}u_1\right]^4 + \sum_{i=3}^{4}\left[\frac{(\boldsymbol{y}_2)_i}{\sigma(\boldsymbol{y}_2)}u_2\right]^4 + \sum_{i\geq 5,(\boldsymbol{y}_2)_i=0}^{n}\left[\frac{(\boldsymbol{y}_1)_i}{\sigma(\boldsymbol{y}_1)}u_1\right]^4 + \sum_{i\geq 5,(\boldsymbol{y}_1)_i=0}^{n}\left[\frac{(\boldsymbol{y}_2)_i}{\sigma(\boldsymbol{y}_2)}u_2\right]^4\right\}$$

$$= \frac{1}{n}\left\{\sum_{i=1}^{2}\frac{\alpha^4}{\frac{1}{4}[1+\frac{4}{n}(\alpha^2-1)]^2}u_1^4 + \sum_{i=3}^{4}\frac{1}{\frac{1}{4}}u_2^4 + \sum_{i\geq 5,(\boldsymbol{y}_2)_i=0}^{n}\frac{1}{\frac{1}{4}[1+\frac{4}{n}(\alpha^2-1)]^2}u_1^4 + \sum_{i\geq 5,(\boldsymbol{y}_1)_i=0}^{n}\frac{1}{\frac{1}{4}}u_2^4\right\}$$

$$= \cdots = 2\left[v_n(\alpha)u_1^4 + u_2^4\right] \quad \text{wobei} \quad v_n(\alpha) = \frac{1+\frac{4}{n}(\alpha^4-1)}{[1+\frac{4}{n}(\alpha^2-1)]^2}$$

Robuste Zielfunktion

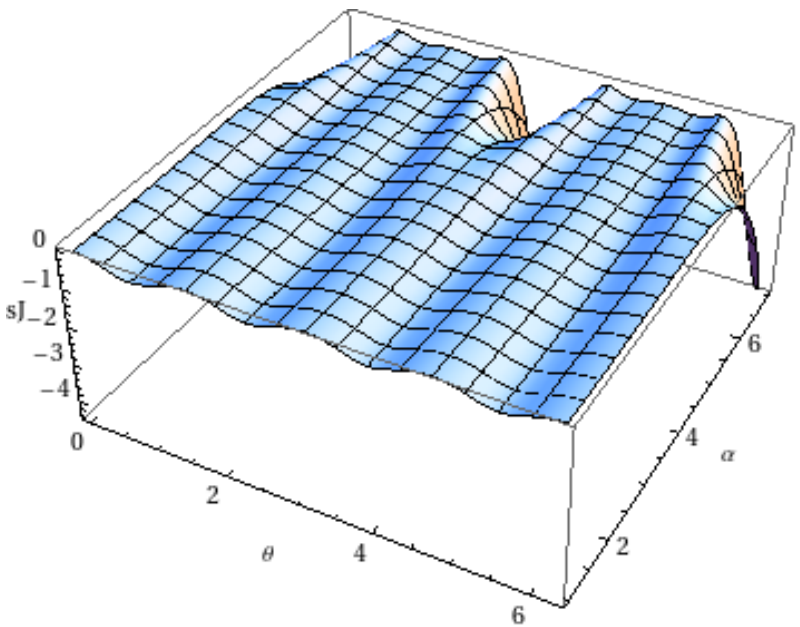
Mit den obigen Rechnungen ist die Wölbung gegeben durch

$$W(Y_s^T \boldsymbol{u}) = M_4(Y_s^T \boldsymbol{u}) - 3M_2^2(Y_s^T \boldsymbol{u}) = 2\left[\upsilon_n(\alpha)u_1^4 + u_2^4 - \frac{3}{2}\right]$$

und die bisher verwendete Zielfunktion ist

$$J(\boldsymbol{u}) = -W^2(Y_s^T \boldsymbol{u}) = -4\left[\upsilon_n(\alpha)u_1^4 + u_2^4 - \frac{3}{2}\right]^2$$

Mit $\boldsymbol{u} = \{\cos(\theta), \sin(\theta)\}$ und $n = 2000$ sieht die Landschaft dieser Zielfunktion so aus:



Ein minimierendes $\boldsymbol{u}$ ist

$$\boldsymbol{u} = \left\{1, \sqrt{\upsilon_n(\alpha)}\right\} / \sqrt{1 + \upsilon_n(\alpha)}$$

wobei $\alpha = 1 \Rightarrow \upsilon_n(1) = 1$ und

$$\boldsymbol{u} \to \boldsymbol{u}^\star \text{ oder } \theta = \tfrac{\pi}{4} \approx 0.8.$$

Jedoch für größeres α liegt das globale Minimum bei $\theta = 0$ oder

$$\boldsymbol{u} \overset{\alpha\to\infty}{\longrightarrow} \{1, 0\}$$

und so werden die Daten nicht gedreht.

Daher ist diese Zielfunktion gegen Ausreißer empfindlich.

Robuste Zielfunktion

Für eine alternative Zielfunktion wird berechnet:

$$M_1(Y_s^T \boldsymbol{u}) = \frac{1}{n}\sum_{i=1}^{n} |\boldsymbol{e}_i^T Y_s^T \boldsymbol{u}| = \frac{1}{n}\sum_{i=1}^{n} \left| \frac{(\boldsymbol{y}_1)_i}{\sigma(\boldsymbol{y}_1)} u_1 + \frac{(\boldsymbol{y}_2)_i}{\sigma(\boldsymbol{y}_2)} u_2 \right|$$

$$= \frac{1}{n}\left\{ \sum_{i=1}^{2} \left| \frac{(\boldsymbol{y}_1)_i}{\sigma(\boldsymbol{y}_1)} u_1 \right| + \sum_{i=3}^{4} \left| \frac{(\boldsymbol{y}_2)_i}{\sigma(\boldsymbol{y}_2)} u_2 \right| \right.$$

$$\left. + \sum_{i\geq 5,(\boldsymbol{y}_2)_i=0}^{n} \left| \frac{(\boldsymbol{y}_1)_i}{\sigma(\boldsymbol{y}_1)} u_1 \right] + \sum_{i\geq 5,(\boldsymbol{y}_1)_i=0}^{n} \left| \frac{(\boldsymbol{y}_2)_i}{\sigma(\boldsymbol{y}_2)} u_2 \right| \right\}$$

$$= \frac{\sqrt{2}}{n}\left\{ \frac{2|\alpha|}{\gamma_n(\alpha)}|u_1| + 2|u_2| + \frac{1}{2}(n-4)\frac{1}{\gamma_n(\alpha)} + \frac{1}{2}(n-4)|u_2| \right\}$$

wobei $\gamma_n(\alpha) = 1 + \frac{4}{n}(\alpha^2 - 1)$,

und mit $\nu_n(\alpha) = [1 + \frac{4}{n}(|\alpha| - 1)]/\gamma_n(\alpha)$,

$$= \frac{1}{\sqrt{2}}[\nu_n(\alpha)|u_1| + |u_2|]$$

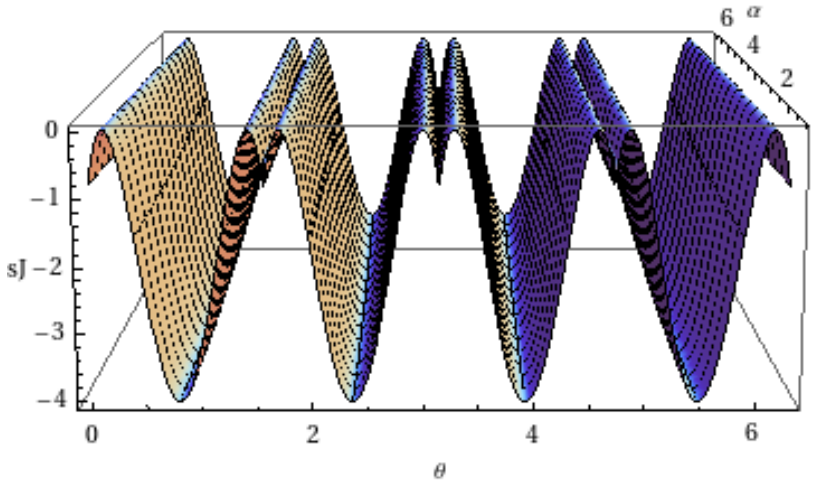
Robuste Zielfunktion

Eine Abtastung $\boldsymbol{n} = \{n_i\}_{i=1}^{n}$ mit der Gaußschen Verteilung $N(\mu, \sigma)$ erfüllt

$$M_1(\boldsymbol{n}) = \frac{1}{n}\sum_{i=1}^{n} |n_i - \mu| \overset{n\to\infty}{\longrightarrow} \sigma\sqrt{\frac{2}{\pi}}$$

Daher wird die folgende Zielfunktion natürlich definiert:

$$J(\boldsymbol{u}) = -\left[M_1(Y_s^{\mathrm{T}}\boldsymbol{u}) - \sqrt{M_2(Y_s^{\mathrm{T}}\boldsymbol{u})}\sqrt{\frac{2}{\pi}}\right]^2 \to -\frac{1}{2}\left[\nu_n(\alpha)|u_1| + |u_2| - \frac{2}{\sqrt{\pi}}\right]^2$$



Mit $\boldsymbol{u} = \{\cos(\theta), \sin(\theta)\}$ und $n = 2000$, sieht die Landschaft dieser Zielfunktion so aus.

Sie wird für größeres α bei $\theta \approx \pi/4$ or $\boldsymbol{u} \approx \boldsymbol{u}^\star$ minimiert.

Daher ist diese Zielfunktion gegen Ausreißer robust.

Optimierung der Robusten Zielfunktion

Im Gegensatz zu den vorherigen Codes ▸ ▸ ▸ wird die robuste Zielfunktion hier mit einer neuen Methode optimiert.

Die Zielfunktion,

$$J(\boldsymbol{u}) = -[M_1(Y_s^T\boldsymbol{u}) - \sqrt{M_2(Y_s\boldsymbol{u})}\sqrt{\tfrac{2}{\pi}}]^2$$
$$= -[M_1(Y_s^T\boldsymbol{u}) - \sqrt{2/\pi}]^2$$

soll unter der Nebenbedingung $\boldsymbol{u}^T\boldsymbol{u} = 1$ minimiert werden.

Die Lösung entspricht einem stationären Punkt der Lagrange Funktion,

$$L(\boldsymbol{u}, \lambda) = J(\boldsymbol{u}) + \lambda(\boldsymbol{u}^T\boldsymbol{u} - 1)/2$$

Die Zielfunktion hat den Gradienten,

$$D_{\boldsymbol{u}}J(\boldsymbol{u}) = -2[M_1(Y_s^T\boldsymbol{u}) - \sqrt{2/\pi}]D_{\boldsymbol{u}}M_1(Y_s^T\boldsymbol{u})$$

wobei der Gradient von $M_1(Y_s^T\boldsymbol{u})$ ist für $\epsilon \ll 1$,

$$D_{\boldsymbol{u}}M_1(Y_s^T\boldsymbol{u}) \approx \frac{1}{n}\sum_{i=1}^{n}[(\boldsymbol{e}_i^T Y_s^T\boldsymbol{u})^2 + \epsilon^2]^{-\frac{1}{2}}(\boldsymbol{e}_i^T Y_s^T\boldsymbol{u})(Y_s\boldsymbol{e}_i)$$

Optimierung der Robusten Zielfunktion

Mit $\phi(\boldsymbol{u}) = 2[M_1(Y_s^T\boldsymbol{u}) - \sqrt{2/\pi}]$ und der SPD Matrix

$$G(\boldsymbol{u}) = \frac{1}{n}\sum_{i=1}^{n}[(\boldsymbol{e}_i^T Y_s^T \boldsymbol{u})^2 + \epsilon^2]^{-\frac{1}{2}} Y_s \boldsymbol{e}_i \boldsymbol{e}_i^T Y_s^T$$

gilt

$$\phi(\boldsymbol{u})G(\boldsymbol{u})\boldsymbol{u} = -D_{\boldsymbol{u}}J(\boldsymbol{u}).$$

Ein stationärer Punkt $(\boldsymbol{u}^\star, \lambda^\star)$ von L erfüllt $-D_{\boldsymbol{u}}J(\boldsymbol{u}^\star) = \lambda^\star \boldsymbol{u}^\star$ oder mit $\lambda^\star = \mu^\star(\boldsymbol{u}^\star)\phi(\boldsymbol{u}^\star)$,

$$G(\boldsymbol{u}^\star)\boldsymbol{u}^\star = \mu^\star(\boldsymbol{u}^\star)\boldsymbol{u}^\star, \quad \boldsymbol{u}^{\star T}\boldsymbol{u}^\star = 1$$

Dieses nicht lineare Eigenraumproblem wird mit der folgenden Vektoriteration gelöst.

Gegeben sei $\boldsymbol{u}_l \approx \boldsymbol{u}^\star$ mit $\|\boldsymbol{u}_l\| = 1$ und eine Aktualisierung $\boldsymbol{u}_{l+1}$ wird so bestimmt:

$$G(\boldsymbol{u}_l)\boldsymbol{u} = \boldsymbol{u}_l, \quad \boldsymbol{u}_{l+1} = \boldsymbol{u}/\|\boldsymbol{u}\|, \quad l = 1, 2, \ldots$$

Optimierung der Robusten Zielfunktion

Nach Konvergenz ist $\boldsymbol{u}^\star = \lim_{l\to\infty} \boldsymbol{u}_l$ die erste Spalte von U^{T}.

Die nächste Spalte von U^{T} wird durch eine modifizierte Vektoriteration bestimmt. Nun sei

$$\tilde{G}(\boldsymbol{u}) = \frac{1}{n}\sum_{i=1}^{n}[(\boldsymbol{e}_i^{\mathrm{T}} Y_{\mathrm{p}}^{\mathrm{T}}\boldsymbol{u})^2 + \epsilon^2]^{-\frac{1}{2}} Y_{\mathrm{p}}\boldsymbol{e}_i\boldsymbol{e}_i^{\mathrm{T}} Y_{\mathrm{p}}^{\mathrm{T}}$$

wobei die projektierten Daten

$$Y_{\mathrm{p}} = (I - \boldsymbol{u}^\star\boldsymbol{u}^{\star\mathrm{T}})Y_{\mathrm{s}}$$

Spalten haben, die von $\boldsymbol{u}^\star$ linear unabhängig sind.

Die modifizierte Vektoriteration ist,

$$[\boldsymbol{u}^\star\boldsymbol{u}^{\star\mathrm{T}} + \tilde{G}(\boldsymbol{u}_l)]\boldsymbol{u} = \boldsymbol{u}_l, \quad \boldsymbol{u}_{l+1} = \boldsymbol{u}/\|\boldsymbol{u}\|, \quad l = 1, 2, \ldots$$

Die verbliebenen Spalten von U^{T} werden ähnlich bestimmt, wobei $\boldsymbol{u}^\star$ oben mit der Matrix $[\boldsymbol{u}_1^\star, \ldots, \boldsymbol{u}_k^\star]$ ersetzt wird, wenn k Spalten $\{\boldsymbol{u}_1^\star, \ldots, \boldsymbol{u}_k^\star\}$ von U^{T} schon berechnet worden sind.

Optimierung der Robusten Zielfunktion

MATLAB Code um *U* mit der obigen Methode zu bestimmen, wobei mit zufälliger Abtastung gestartet wird.

```
itmax = 1000; lmax = 100; tol = 1.0e-6;
I = eye(m); U = [];
Ms = sqrt(2/pi); ep = 1.0e-3;
for i=1:(m-1)
    if (i == 1)
      Yp = Ys;
    else
      Yp = (I - U'*U)*Ys;    % Daten projizieren
    end
    for l=1:lmax          % Zufaellige Abtastung
      u = randn(m,1);
      if (i > 1)
        u = (I - U'*U)*u;
      end
      u = u/norm(u);
```

Optimierung der Robusten Zielfunktion

```
      M1 = mean(sqrt((u'*Yp).^2+ep^2));
      J = -(M1 - Ms)^2;
      if (l == 1)
        Jm = J; ua = u;
      else
        if (J < Jm) Jm = J; ua = u; end
      end
    end
    u = ua;
    yy = zeros(m,m,n);     % Vorrechnung fuer G
    for k=1:n
      yy(:,:,k) = Yp(:,k)*Yp(:,k)';
    end
    for it=1:itmax         % Vektoriteration
      ua = u;
      Lp = 1./sqrt((u'*Yp).^2 + ep);
      yL = reshape(yy,m*m,n)*spdiags(Lp',0,n,n);
      G = mean(reshape(yL,m,m,n),3);
```

Optimierung der Robusten Zielfunktion

```
        if (i == 1)
          u = G \ ua;
        else
          u = (G + U'*U) \ ua;
          u = (I - U'*U)*u;
        end
        u = u/norm(u);
        u = sign(dot(u,ua))*u;
        du = (I - ua*ua')*u;
        if (max(abs(du))< tol) break; end
    end
    U = [U;u'];
end
u = randn(m,1);          % letzte Reihe von U
u = (I - U'*U)*u;
u = u/norm(u);
U = [U;u'];
Xz = U*Ys;
```

Formulierung im Funktionenraum

Analog zur diskreten Formulierung ▸ seien $\{z_i(\omega) : \omega \in \Omega\}_{i=1}^{\infty}$ abzählbar viele Zufallsfunktionen (z.B. Bilder, vgl. ▸), die erfüllen,

$$\int_{\Omega} |z_i(\omega)|^2 d\omega < \infty \quad \text{oder} \quad z_i \in L^2(\Omega)$$

Sei $\{e_j(\omega)\}_{j=1}^{\infty}$ eine natürliche orthonormale Basis für $L^2(\Omega)$,

$$\int_{\Omega} e_i(\omega) e_j(\omega) d\omega = \delta_{i,j}$$

sodass die Zufallsfunktionen sich bezülich der Basis so darstellen lassen,

$$z_i(\omega) = \sum_{j=1}^{\infty} z_{i,j} e_j(\omega), \quad z_{i,j} = \int_{\Omega} z_i(\omega) e_j(\omega) d\omega$$

Die statistischen Eigenschaften der Zufallsfunktionen lassen sich bezüglich der Koeffizienten $\{z_{i,j}\}$ formulieren:

- $z_1(\omega)$ sei normal verteilt wenn $\{z_{1,j}\}_{j=1}^{N}$, $\forall N$, normal verteilt sind,
- $z_1(\omega)$ und $z_2(\omega)$ seien unabhängig wenn $\{z_{1,j}\}_{j=1}^{N}$ und $\{z_{2,j}\}_{j=1}^{N}$, $\forall N$, unabhängig sind.

Formulierung im Funktionenraum

Angenommen sind die Zufallsfunktionen $\{z_i(\omega) : \omega \in \Omega\}_{i=1}^{\infty}$ paarweise unabhängig und nicht Gauß verteilt.

Wenn die Zufallsfunktionen als Bilder interpretiert werden, und die pixelweise Darstellung verwendet wird ▸, nimmt man

$$e_i(\omega) = \begin{cases} 1, & \omega \in \text{Pixel}_i \\ 0, & \text{sonst} \end{cases} \qquad \text{Pixel}_i = [\omega_i, \omega_i + h]$$

aber so eine orthonormale Basis ist notwendigerweise endlich.

Seien $\{z_i(\omega, t) : \omega \in \Omega\}_{i=1}^{\infty}$ über $0 \leq t \leq T$ kontinuierlich abgetastete Realisierungen der Zufallsfunktionen. Diese sind die Quellen, die unabhängig und nicht Gauß verteilt sind.

Da die Quellen unabhängig sind, sind sie auch unkorreliert ▸:

$$\delta_{i,j} = \frac{1}{T} \int_0^T \left[\int_\Omega z_i(\omega, t) z_j(\omega, t) d\omega \right] dt =$$

$$\frac{1}{T} \int_0^T \left[\sum_{k,l=1}^{\infty} z_{i,k}(t) z_{j,l}(t) \underbrace{\int_\Omega e_k(\omega) e_l(\omega) d\omega}_{=\delta_{k,l}} \right] dt = \sum_{k=1}^{\infty} \frac{1}{T} \int_0^T z_{i,k}(t) z_{j,k}(t) dt$$

Formulierung im Funktionenraum

Seien $\{y_i(\omega, t) : \omega \in \Omega\}_{i=1}^{\infty}$ über $0 \leq t \leq T$ kontinuierlich abgetastete Messungen, wobei für unbekannte $A = \{a_{i,j}\}_{i,j=1}^{\infty}$

$$y_i(\omega, t) = \sum_{j=1}^{\infty} a_{i,j} z_j(\omega, t)$$

Seien diese Daten so zentriert,

$$y_i^z(\omega, t) = y_i(\omega, t) - \mu(y_i(\omega)), \quad \mu(y_i(\omega)) = \frac{1}{T} \int_0^T y_i(\omega, t) dt$$

und definiere die Kovarianzen

$$\kappa_{i,j} = \frac{1}{T} \int_0^T \left[\int_\Omega y_i^z(\omega, t) y_j^z(\omega, t) d\omega \right] dt$$

Angenommen hat der Operator $K : L^2(\Omega) \to L^2(\Omega)$

$$[Kf](\omega) = \sum_{i=1}^{\infty} \left[\sum_{j=1}^{\infty} \kappa_{i,j} f_j \right] e_i(\omega), \quad f_j = \int_\Omega f(\omega) e_j(\omega)$$

eine orthonormale Basis von Eigenfunktionen $\{v_k(\omega)\}_{k=1}^{\infty}$ mit positiven Eigenwerten $\lambda_1 > \lambda_2 > \cdots > \lambda_k \overset{k \to \infty}{\longrightarrow} 0$, d.h.

Formulierung im Funktionenraum

es gelten

$$\lambda_k v_k(\omega) = [K v_k](\omega), \qquad \int_\Omega v_k(\omega) v_l(\omega) d\omega = \delta_{k,l}, \quad k, l = 1, 2, \ldots$$

oder mit

$$v_k(\omega) = \sum_{i=1}^{\infty} e_i(\omega) v_{i,k}$$

folgen

$$v_{i,k} \lambda_k = \sum_{j=1}^{\infty} \kappa_{i,j} v_{j,k}, \qquad \sum_{i=1}^{\infty} v_{i,k} v_{i,l} = \delta_{k,l}, \quad k, l = 1, 2, \ldots$$

Die zentrierten Daten lassen sich so darstellen,

$$y_i^z(\omega, t) = \sum_{k=1}^{\infty} y_{i,k}^z(t) e_k(\omega), \quad y_{i,k}^z(t) = \int_\Omega y_i^z(\omega, t) e_k(\omega) d\omega$$

und es folgt

$$v_{i,k} \lambda_k = \sum_{j=1}^{\infty} \kappa_{i,j} v_{j,k} =$$

$$\sum_{j=1}^{\infty} \left\{ \frac{1}{T} \int_0^T \left[\sum_{m,n=1}^{\infty} y_{i,m}^z(t) y_{j,n}^z(t) \underbrace{\int_\Omega e_m(\omega) e_n(\omega) d\omega}_{=\delta_{m,n}} \right] dt \right\} v_{j,k}$$

Formulierung im Funktionenraum

oder

$$v_{i,k}\lambda_k = \sum_{j=1}^{\infty}\left\{\frac{1}{T}\int_0^T \sum_{m=1}^{\infty} y^z_{i,m}(t)y^z_{j,m}(t)dt\right\} v_{j,k}$$

Definiere die gesphärten Daten

$$y^s_i(\omega,t) = \sum_{k=1}^{\infty} y^s_{i,k}(t)e_k(\omega) \quad \text{wobei} \quad y^s_{i,k}(t) = \lambda_i^{-\frac{1}{2}} \sum_{m=1}^{\infty} v_{m,i}y^z_{m,k}(t)$$

die unkorreliert sind,

$$\frac{1}{T}\int_0^T \left[\int_\Omega y^s_i(\omega,t)y^s_j(\omega,t)d\omega\right] dt =$$

$$\frac{1}{T}\int_0^T \sum_{k,l=1}^{\infty} y^s_{i,k}(t)y^s_{j,l}(t)\underbrace{\left[\int_\Omega e_k(\omega)e_l(\omega)d\omega\right]}_{=\delta_{k,l}} dt = \frac{1}{T}\int_0^T \sum_{k=1}^{\infty} y^s_{i,k}(t)y^s_{j,k}(t)dt$$

$$= \frac{1}{\sqrt{\lambda_i\lambda_j}}\sum_{m=1}^{\infty} v_{m,i}\underbrace{\sum_{n=1}^{\infty}\left\{\frac{1}{T}\int_0^T \sum_{k=1}^{\infty} y^z_{m,k}(t)y^z_{n,k}(t)dt\right\} v_{n,j}}_{=v_{m,j}\lambda_j} = \underbrace{\sqrt{\frac{\lambda_j}{\lambda_i}}\sum_{m=1}^{\infty} v_{m,i}v_{m,j}}_{=\delta_{i,j}}$$

Formulierung im Funktionenraum

Nun wird eine orthonormale Basis $\{u_k(\omega)\}_{k=1}^{\infty}$ für $L^2(\Omega)$ gesucht,

$$u_k(\omega) = \sum_{i=1}^{\infty} e_i(\omega) u_{i,k} \quad \delta_{k,l} = \int_{\Omega} u_k(\omega) u_l(\omega) d\omega =$$

$$\sum_{i,j=1}^{\infty} u_{i,k} u_{j,k} \underbrace{\int_{\Omega} e_i(\omega) e_j(\omega) d\omega}_{=\delta_{i,j}} = \sum_{i=1}^{\infty} u_{i,k} u_{j,k}$$

wobei mit

$$x_i^z(\omega, t) = \sum_{k=1}^{\infty} x_{i,k}^z(t) e_k(\omega)$$

$$x_{i,k}^z(t) = \sum_{n=1}^{\infty} u_{i,n} y_{n,k}^s(t) = \sum_{n=1}^{\infty} u_{i,n} \lambda_n^{-\frac{1}{2}} \sum_{m=1}^{\infty} v_{m,n} y_{m,k}^z(t)$$

die Daten $\{x_i^z(\omega, t) : \omega \in \Omega\}_{i=1}^{\infty}$, die wegen der Eigenschaften von $\{u_k(\omega)\}_{k=1}^{\infty}$ schon unkorreliert sind, auch paarweise möglichst unabhängig sind.

Somit wird A^{-1} mit W approximiert, wobei $W = \{w_{i,j}\}_{i,j=1}^{\infty}$ und

$$w_{i,j} = \sum_{n=1}^{\infty} u_{i,n} \lambda_n^{-\frac{1}{2}} v_{j,n}$$

Formulierung im Funktionenraum

Basierend auf das obige Inversionsformel bezeichne mit

$$x_i^z(u_i) = \sum_{k=1}^{\infty} \left[\sum_{n=1}^{\infty} u_{i,n} y_{n,k}^s(t) \right]$$

die Abhängigkeit von $u_i = \{u_{i,n}\}_{n=1}^{\infty}$ und definiere

$$M_2(x_i^z(u_i)) = \frac{1}{T} \int_0^T \left[\sum_{n=1}^{\infty} u_{i,n} y_{n,k}^s(t) \right]^2 dt$$

und

$$M_e(x_i^z(u_i)) = \frac{1}{T} \int_0^T \exp \left[- \left(\sum_{n=1}^{\infty} u_{i,n} y_{n,k}^s(t) \right)^2 \right] dt$$

Um u_i zu bestimmen, ist die folgende robuste Zielfunktion für jedes i zu minimieren:

$$J(u_i) = - \left[M_e(x_i^z(u_i)) - 1/\sqrt{1 + 2M_2(x_i^z(u_i))} \right]^2$$

unter der Nebenbedingung

$$\sum_{n=1}^{\infty} u_{i,n} u_{j,n} = \delta_{i,j}$$