Testen der Gleichheit von räumlichen ACF: Eine erste Implementierung

Carsten Stahl

2025-03-17

Einführung

Dieses Dokument baut auf dem Teststatistik aus Jentsch and Pauly (2015), welche auf die Gleichheit von zwei (oder mehr) Spektraldichten (und somit auch ACFs) testet. Im genauen werden die folgenden Hypothesen getestet:

$$H_0 : f_X(\omega) = f_Y(\omega)$$
$$H_1 : \exists A \in \mathcal{B}([-\pi, \pi]) | \lambda(A) > 0 : f_X(\omega) \neq f_Y(\omega) \quad \forall \omega \in A$$

Verwendet wird hierbei eine L_2 Teststatistik:

$$T = L_2(\hat{f}_X, \hat{f}_Y) = \int_{-\pi}^{\pi} (\hat{f}_X(\omega) - \hat{f}_Y(\omega))^2 d\omega$$

Jentsch and Pauly (2015) verbessern die Güte des Tests für kleinere Stichproben durch Resampling-Techniken, welche am Ende mehrere Werte für T erzeugen.

Der Beitrag besteht darin, die Teststatistik auf den *räumlichen* Fall anzuwenden. Da dies ein erster Versuch ist, mit dem Problem vertraut zu werden, schauen wir uns hier den einfachsten Fall an. Wir sehen von mehrdimensionalen stochastischen Prozessen ab und schauen uns lediglich zwei eindimensionale räumliche schwach stationäre stochastische Prozesse an. Es soll außerdem nur auf Gleichheit getestet werden und keine Brücke zu komplexeren Hypothesen geschlagen werden. Die Implementierung dieses Falles wurde absichtlich simpel gehalten. Performantere und allgemeinere Implementierungen werden im späteren Verlauf folgen.

Kurze Zusammenfassung der Ergebnisse

Trotz der vereinfachenden Annahmen war die Implementierung technisch anspruchsvoll. Getestet wurde das Verfahren auf zwei verschienden Prozessen - einer räumlichen Verallgemeinerung des $MA(q)$-Prozesses und einem Prozess basierend auf einer einfachen Spektraldichte. Ablehnung und Annahme der H_0 Hypothese geschah unter beiden Prozessen willkührlich, egal ob die Daten unter H_0 oder H_1 erzeugt wurden. Diese Ergebnisse deuten stark auf einen Fehler meinerseits hin. Das Verfahren skalliert in $\mathcal{O}(NMCB)$ auf einem $N \times M$ Gitter, wobei C die Anzahl der Funktionsaufrufe für die numerische Integration ist und B die Anzahl der Neuberechnungen von T_n^*.

Die weiteren Sektionen enthalten zuerst eine theoretische Erweiterung von der Teststatistik auf den räumlichen Fall und anschließend die Implementierung.

Formelle Beschreibung des Problems

Es seien $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum, $(\mathbb{R}, \mathcal{B})$ ein Maßraum, und $s \in I = \mathbb{Z}^2$ eine räumliches Gitter. Nun seien $X_s : \Omega \to \mathbb{R}$ und $Y_s : \Omega \to \mathbb{R}$ räumliche Zufallsvariablen (sprich $\mathcal{A} - \mathcal{B}$ messbar).

Dazu seien $\mu_i : I \to \mathbb{R}$ und $C_i : I^2 \to \mathbb{R}$ die Mittelwert- und Autokovarianzfunktionen für $i \in \{X, Y\}$. Wir gehen von *schwach stationären* Prozessen $(X_s)_{s\in I}$ und $(Y_s)_{s\in I}$ aus. Das heißt es gilt für μ_i und C_i:

$$\begin{aligned} \mu_i(s) &= C \quad \forall s \in I \\ C_i(s,t) &= C_i(s+h, t+h) \quad \forall s,t,h \in I \end{aligned}$$

Es wird jetzt eine Realisierung oder Pfad für ein Rechteck $I_{N,M}$ des räumlichen Gitters I beobachtet. Formal würden wir ohne Einschränkung der Allgemeinheit $I_{N,M}$ folgendermaßen definieren:

$$I_{N,M} := \{(i,j) | i,j \in \mathbb{Z} \wedge 0 \leq i < N \wedge 0 \leq j < M\}$$

Wir werden mit diesem Pfad eine Spektraldichte mit dem *smoothed Periodogram* schätzen und darauf die Teststatistik aufbauen.

Schätzung der Spekraldichten

Für das Periodogramm wenden wir die diskrete Fourier-Transformation auf die Beobachtungen an:

$$I_y(\omega_1, \omega_2) = \frac{1}{(2\pi)^2 NM} \left| \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} y_{ij} \exp(-i(\omega_1 k + \omega_2 l)) \right|^2 \quad \text{(für } x_{ij} \text{ analog)}$$

Wichtig ist hier, dass man sich Paare von Frequenzen anschaut - eine für die Spalte (oder y-Koordiante) und eine für die Zeile (oder x-Koordiante). Anschließend werden die Werte mit einem Kernel geglättet, um die finale Schätzung zu erhalten:

$$\hat{f}_y(\omega_1, \omega_2) := \frac{1}{NM} \sum_{k=-\lfloor (N-2)/2 \rfloor}^{\lfloor N/2 \rfloor} \sum_{l=-\lfloor (M-2)/2 \rfloor}^{\lfloor M/2 \rfloor} K_h(\omega_1 - \omega_k, \omega_2 - \omega_l) I_y(\omega_k, \omega_l) \quad \text{(für } x \text{ analog)}$$

Wobei $K_h(\omega_1, \omega_2)$ der Kernel, unter Verwendung der Bandbreite h, für die Glättung ist. h ist hier eine zweidimensionale Größe, da wir über zwei verschiedene Frequenzen glätten. Intuitiv lässt sich bereits sagen, dass ein verhältnismäßig kleines h zu mehr Varianz in der Schätzung führt und bei einem großen h evtl. schmale Modi (erzeugt durch ganz bestimmte Signale) verloren gehen. Was passiert, wenn das Verhältniss zwischen den Bandbreiten größer wird, bleibt offen.

Teststatistik: T_n

Jentsch and Pauly (2015) beschreiben ihre Teststatistik ohne resampling wiefolgt:

$$\begin{aligned} T_n &:= n\sqrt{h} \int_{-\pi}^{\pi} \sum_{r=1}^{q} \left\| \frac{1}{n} \sum_{k=-\lfloor (n-1)/2 \rfloor}^{\lfloor n/2 \rfloor} K_h(\omega - \omega_k) I_{rr}(\omega_k) - \tilde{I}(\omega) \right\|^2 d\omega \\ &= n\sqrt{h} \int_{-\pi}^{\pi} \sum_{r=1}^{q} \|\hat{f}_{rr}(\omega) - \tilde{f}(\omega)\|^2 d\omega \end{aligned}$$

Wobei $\tilde{f}(\omega) = \frac{1}{q} \sum_{j=1}^{q} \hat{f}_{jj}(\omega)$ der Durchschnitt der geschätzten Spektraldichten ist.
Dank unserer vereinfachten Annahmen können wir die Teststatistik drastisch vereinfachen und anschließend

auf unseren räumlichen Fall anpassen. Statt der quadrierten euklidischen Norm lässt sich der quadriete Abstand verwenden. Wir benutzen zwei räumliche Prozesse, also wird $q = 2$ gesetzt. Vereinfacht heißt das:

$$T_n := n\sqrt{h} \int_{-\pi}^{\pi} (\hat{f}_X(\omega) - \tilde{f}(\omega))^2 + (\hat{f}_Y(\omega) - \tilde{f}(\omega))^2 d\omega$$

Mit $\tilde{f}(\omega) := \frac{1}{2}\left[\hat{f}_X(\omega) + \hat{f}_Y(\omega)\right]$ und $n = NM$.

Die Riemann Summe von T_n ist:

$$T_n \approx \sqrt{h} 2\pi \sum_{k=-\lfloor -(n-1)/2 \rfloor}^{\lfloor n/2 \rfloor} (\hat{f}_X(\omega_k) - \tilde{f}(\omega_k))^2 + (\hat{f}_Y(\omega_k) - \tilde{f}(\omega_k))^2$$

Würden wir jetzt den räumlichen Aspekt berücksichtigen, müssten wir die Schätzungen der räumlichen Spektraldichten einsetzen und über beide Frequenzen integrieren:

$$T_n^s := n\sqrt{h} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} (\hat{f}_X(\omega, \tilde{\omega}) - \tilde{f}(\omega, \tilde{\omega}))^2 + (\hat{f}_Y(\omega, \tilde{\omega}) - \tilde{f}(\omega, \tilde{\omega}))^2 d\omega d\tilde{\omega}$$

Und über die Riemann summe approximieren:

$$T_n^s \approx \sqrt{h} 4\pi^2 \sum_{k=\lfloor -(N-1)/2 \rfloor}^{\lfloor N/2 \rfloor} \sum_{l=\lfloor -(M-1)/2 \rfloor}^{\lfloor M/2 \rfloor} (\hat{f}_X(\omega_k, \omega_l) - \tilde{f}(\omega_k, \omega_l))^2 + (\hat{f}_Y(\omega_k, \omega_l) - \tilde{f}(\omega_k, \omega_l))^2$$

Teststatistik: T_n^*

Die Teststatistik T_n^* ist der randomisierte Teil der Teststatistik. Randomisiert in dem Sinne, dass **vor** der Schätzung der Spektraldichten, die Werte der zwei Periodogramme an den Fourier-Frequenzen zufällig getauscht werden. Jentsch and Pauly (2015) definieren dieses zufällige Tauschen als Gleichverteilung über den Raum aller möglichen Permutationen $\pi_k(q)$, wobei k der zugehörige Index zu der Fourier-Frequenz und q der Zufallsprozess ist. Aber für unsere zwei Zufallsprozesse reicht eine Bernoulli Verteilung $B_{kl} \sim Bernoulli(0.5)$ für das Paar der Fourier-Frequenzen ω_k und ω_l. Die neue Definition von T_n^* ist somit:

$$
\begin{aligned}
T_n^* =& n\sqrt{h}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \\
&\left|\frac{1}{n}\sum_{k=-\lfloor (N-1)/2\rfloor}^{\lfloor N/2\rfloor}\sum_{l=-\lfloor (M-1)/2\rfloor}^{\lfloor M/2\rfloor} K_h(\omega-\omega_k,\tilde{\omega}-\omega_l)(B_{kl}I_y(\omega_k,\omega_l)+(1-B_{kl})I_x(\omega_k,\omega_l)-\tilde{I}(\omega_k,\omega_l))\right|^2 \\
&+\left|\frac{1}{n}\sum_{k=-\lfloor (N-1)/2\rfloor}^{\lfloor N/2\rfloor}\sum_{l=-\lfloor (M-1)/2\rfloor}^{\lfloor M/2\rfloor} K_h(\omega-\omega_k,\tilde{\omega}-\omega_l)((1-B_{kl})I_y(\omega_k,\omega_l)+B_{kl}I_x(\omega_k,\omega_l)-\tilde{I}(\omega_k,\omega_l))\right|^2 \\
&d\omega d\tilde{\omega} \\
\approx& \sqrt{h}4\pi^2 \sum_{\tilde{k}=\lfloor -(N-1)/2\rfloor}^{\lfloor N/2\rfloor}\sum_{\tilde{l}=\lfloor -(M-1)/2\rfloor}^{\lfloor M/2\rfloor} \\
&\left|\frac{1}{n}\sum_{k=-\lfloor (N-1)/2\rfloor}^{\lfloor N/2\rfloor}\sum_{l=-\lfloor (M-1)/2\rfloor}^{\lfloor M/2\rfloor} K_h(\omega_{\tilde{k}}-\omega_k,\omega_{\tilde{l}}-\omega_l)(B_{kl}I_y(\omega_k,\omega_l)+(1-B_{kl})I_x(\omega_k,\omega_l)-\tilde{I}(\omega_k,\omega_l))\right|^2 \\
&+\left|\frac{1}{n}\sum_{k=-\lfloor (N-1)/2\rfloor}^{\lfloor N/2\rfloor}\sum_{l=-\lfloor (M-1)/2\rfloor}^{\lfloor M/2\rfloor} K_h(\omega_{\tilde{k}}-\omega_k,\omega_{\tilde{l}}-\omega_l)I_y(\omega_k,\omega_l)+B_{kl}I_x(\omega_k,\omega_l)-\tilde{I}(\omega_k,\omega_l))\right|^2 \\
&d\omega d\tilde{\omega}
\end{aligned}
$$

Resampling Verfahren

Jentsch and Pauly (2015) beschreiben ein unzentriertes und etwas schnelleres Testverfahren φ_n^* zu dem Signifikanzniveau α mit den folgenden Schritten (hier abgekürzt mit "S."):

S. 1 Berechne T_n für den beobachteten Pfade $\{x_{11},...,x_{NM}\}$ und $\{y_{11},...,y_{NM}\}$.

S. 2 Ziehe B_{kl} neu für alle $k \in \{1,...,\lfloor \frac{N}{2} \rfloor\}$ und $l \in \{1,...,\lfloor \frac{M}{2} \rfloor\}$

S. 3 Berechne $T_n^*(b)$ neu

S. 4 Wiederhole Schritt 2 und 3 $B-1$ mal

S. 5 Die finale Teststatistik ist: $T := \frac{1}{B}\sum_{b=1}^{B} 1\{T_n > T_n^{*(b)}\}$. Lehne H_0 ab, wenn $T > \alpha$.

Wir werden im folgenden unsere Implementierung auf diesen Schritten aufbauen.

Implementierung

Simulation `gridMA(N, M, K)`

Um unsere Implementierung validieren zu können, brauchen wir Daten, welche sowohl unter H_0 als auch unter der Alternative erzeugt wurden. Wie bereits in der Einleitung erwähnt, wollen wir alles so simpel wie möglich halten. Der Prozess ist daher eine einfache Linearkombination von normalverteilten Variablen und stark von der Simulation eines $MA(q)$ Prozesses inspiriert. Der Grundgedanke hierbei ist, dass wir die Koeffizienten φ_k des Lag-Polynoms reinterpretieren von "linearer Einfluss von Residuen zu Lag k" zu "linearer Einfluss von Residuen zum Abstand k". Also heißt dass für X_{ij}, dass jedes Residuum mit Abstand k einen linearen Einfluss von φ_k auf X_{ij} nimmt. Dieser Prozess lässt sich mit einer 2D Konvolution umsetzen und ist nicht nur stationär, sondern hat auch eine isotrope Kovarianzfunktion [1].

[1] Diese Eigenschaften halten auch für den raumzeitlichen Fall und lassen sich dann mit einer 3D Konvolution umsetzen.

Im genaueren seien zuerst $\{\varepsilon_s\}_{s\in I_{\tilde{N}\tilde{M}}}$ mit $\varepsilon_s \overset{iid}{\sim} \mathcal{N}(0,1)$. Anschließend sei $K \in \mathbb{R}^{3\times 3}$ der Kernel der 2D Konvolution:

$$K := \begin{pmatrix} \varphi_1 & \varphi_1 & \varphi_1 \\ \varphi_1 & 1 & \varphi_1 \\ \varphi_1 & \varphi_1 & \varphi_1 \end{pmatrix}$$

$$X_{ij} := \left[\begin{pmatrix} \varepsilon_{11} & \cdots & \varepsilon_{1\tilde{M}} \\ \vdots & \vdots & \vdots \\ \varepsilon_{\tilde{N}1} & \cdots & \varepsilon_{\tilde{N}\tilde{M}} \end{pmatrix} * \begin{pmatrix} \varphi_1 & \varphi_1 & \varphi_1 \\ \varphi_1 & 1 & \varphi_1 \\ \varphi_1 & \varphi_1 & \varphi_1 \end{pmatrix} \right]_{ij} = \varphi_1 \sum_{k=1}^{3}\sum_{l=1}^{3} \varepsilon_{(i+k-2)(j+l-2)} + (1-\varphi_1)\varepsilon_{ij}$$

Aus diesem stochastischen Prozess ergibt sich nun die Mittelwertfunktion:

$$\begin{aligned} \mathbb{E}(X_s) &= \mathbb{E}\left(\varphi_1 \sum_{k=1}^{3}\sum_{l=1}^{3} \varepsilon_{(i+k-2)(j+l-2)} + (1-\varphi_1)\varepsilon_{ij} \right) \\ &= \varphi_1 \cdot 0 + (1-\varphi_1)\cdot 0 = 0 \quad \forall s \in I_{NM} \end{aligned}$$

Für die Kovarianzfunktionen definieren wir die Hilfsmenge $B(s,d)$, welche alle für ein gegebenes $s = (s_1, s_2) \in I_{NM}$ aus dem Gitter alle Elemente des Gitters enthält, welche L_1 Distanz d zu s haben:

$$B(s,d) := \{(c,e) \in I_{NM} : |c - s_1| + |e - s_2| = d\}$$

Mit B lässt sich die Kovarianz schreiben als:

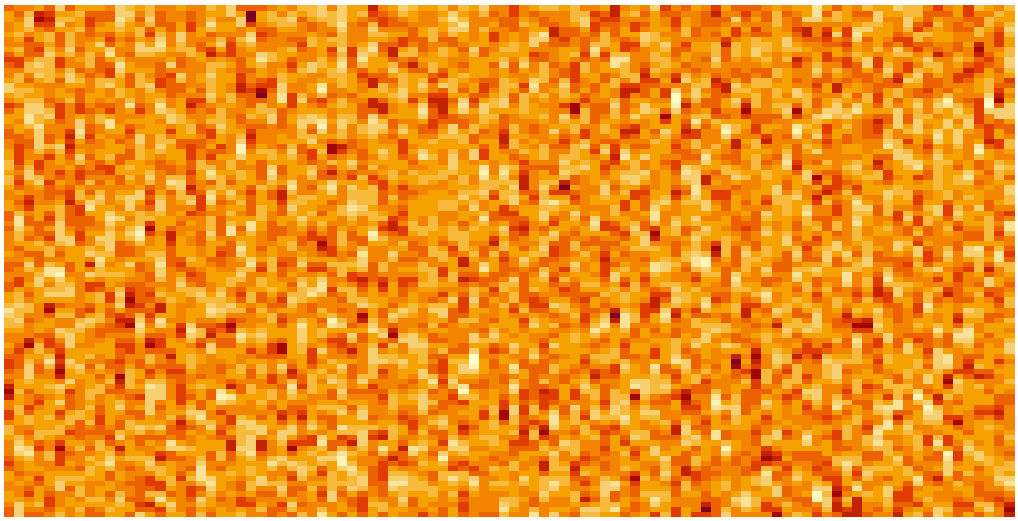
$$\begin{aligned} Cov(X_s, X_t) &\underset{Cov(\varepsilon_i,\varepsilon_j)=0 \;\; i\neq j}{=} \varphi_1^2 \sum_{j\in[B(s,1)\cap B(t,1)]} Var(\varepsilon_j) + \varphi_1 1_{B(s,1)}(t) Var(\varepsilon_t) + \varphi_1 1_{B(t,1)}(s) Var(\varepsilon_s) \\ &\underset{Var(\varepsilon_i)=1}{=} \varphi_1^2 \#(B(s,1)\cap B(t,1)) + \varphi_1 1_{B(t,1)}(s) + \varphi_1 1_{B(s,1)}(t) \\ &\underset{1_{B(s,1)}(t)=1_{B(t,1)}(s)}{=} \varphi_1^2 \#(B(s,1)\cap B(t,1)) + 2\varphi_1 1_{B(t,1)}(s) \end{aligned}$$

Da hier die Werte am Ende nur noch von der Distanz zwischen s und t abhängig sind, ist die Kovarianz ebenfalls nur von der Distanz abhängig. Daraus ergibt sich, dass $C(s,t) = Cov(X_s, X_t)$ ein isotrop ist.

Die Implementierung mit $\varphi_1 = 0.5$ wurde mithilfe von `convolve_image` Funktion aus dem `ravetools` Package umgesetzt. Der Parameter `padding` dient dazu, sicherzustellen, dass auch den Randelementen von $\{X_s\}_{s\in I_{NM}}$ alle Residuen zur Verfügung stehen:

```
gridMA <- function(N, M, K, padding = nrow(K) %/% 2) {
  N_tilde <- N + 2 * padding
  M_tilde <- M + 2 * padding
  n <- N_tilde * M_tilde

  eps <- matrix(rnorm(n),
                nrow = M_tilde,
                ncol = N_tilde)

  x <- convolve_image(eps, K)
  # cut padding
  return(x[(padding+1):(M+padding),(padding+1):(N+padding)])
}

K <- matrix(0, nrow = 3, ncol = 3)
```

```r
K[2,] <- .5
K[2,2] <- 1

image(gridMA(100, 100, K), axes = F, main = "Pfad des Gitterprozesses")
```



Periodogramm `I(x)`

`R` bietet eine breits implementierte 2d diskrete Fourier-Transformation in `fft`.

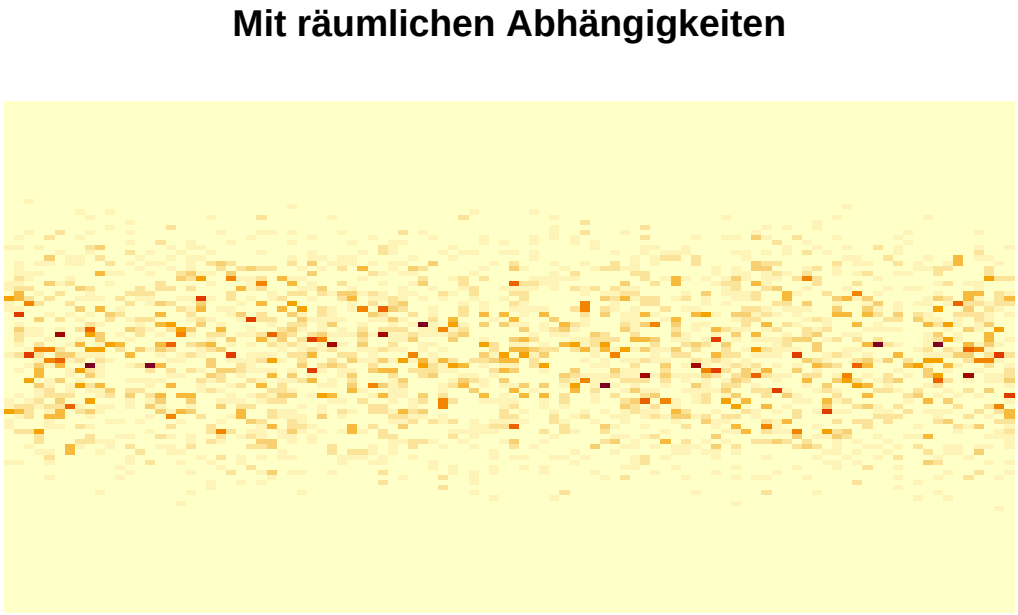
Um aus dieser Fourier-transformation jetzt das Periodogramm zu bekommen, müssen wir nur noch die Modi der Einträge nehmen und anschließend normieren. Daraus ergibt sich:

$$I(\omega_1, \omega_2) = \frac{1}{\pi^2 NM} \text{abs}(fft_y(\omega_1, \omega_2))^2$$

Abschließend müssen die Frequenzen von $[0, 2\pi]$ auf $[-\pi, \pi]$ versetzt werden unter Benutzung des fft-shifts.

```r
I <- function(x){
  N <- nrow(x)
  M <- ncol(x)

  # apply scaling
  res <- (1/(pi^2*N*M))*Mod(fft(x))^2

  # fft shift such that 0 is at the center and nyquist is at the borders
```

```
  return(res[
    c(
      ((N %/% 2)+1):N,
      1:(N %/% 2)),
    c(
      ((M %/% 2)+1):M,
      1:(M %/% 2))]
    )
}

image(I(gridMA(100, 100, K)), axes = F, main = "Mit räumlichen Abhängigkeiten")
```



Erstaunlicherweise scheint es in unserem Prozess viele Signale mit langen Perioden zu geben und fast keine mit sehr kurzen Perioden, obwohl jedes X_s nur von Residuen aus der direkten Umgebung von s beeinflusst wird.

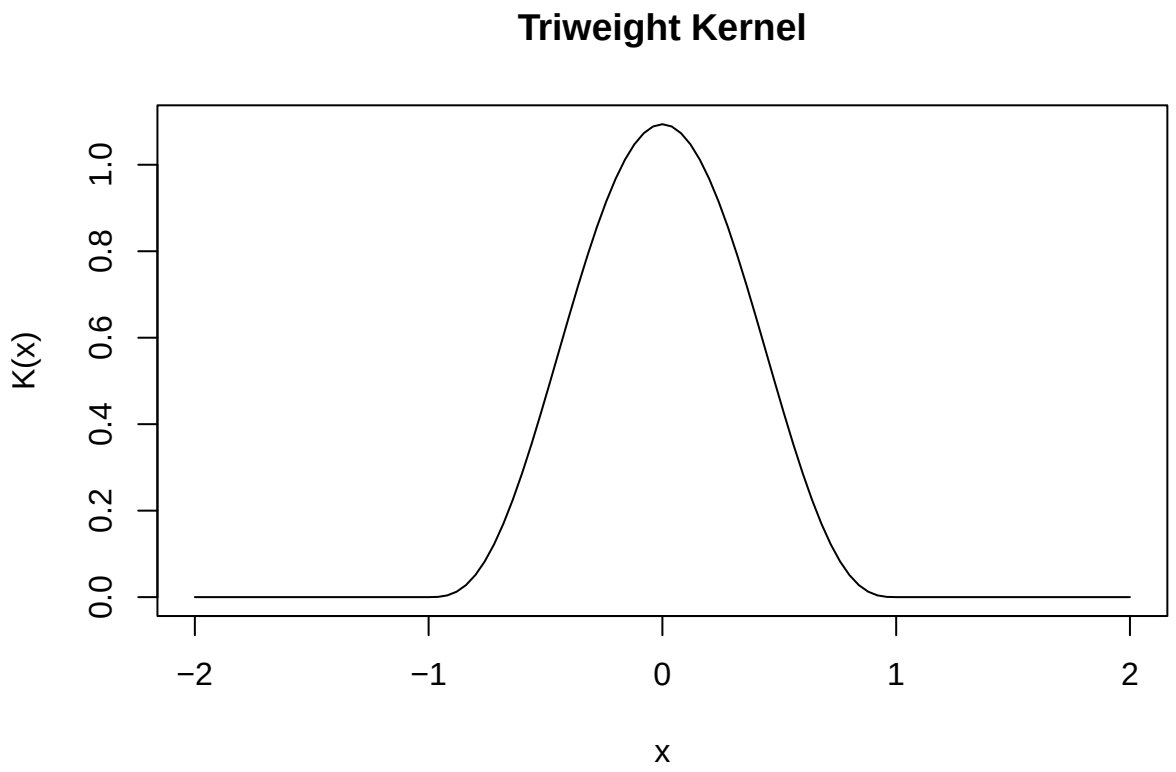
Smoothed Periodogram `I_smooth(I, kernel)`

Für die wie bereits erwähnt brauchen wir für eine konsistente Schätzung der Spektraldichte ein smoothed Periodogram. Im Grunde handelt es sich hierbei um eine Kerndichteschätzung eines Periodogramms. Aus diesem Grund brauchen wir zuerst einen Kernel. Der Kernel dieses Verfahrens ist der Triweight Kernel:

$$K_h(u) = \frac{35}{32}\left(1 - \frac{u}{h}\right)^3$$

Die Wahl war unbegründet und lässt noch Raum zum Experimentieren.

```r
# implementing triweight kernel
tr_k <- function(u, h = 1) ifelse(u/h <= 1 & u / h >= -1, (35 / 32) * (1-(u/h)^2)^3, 0)

curve(tr_k, -2, 2, main = "Triweight Kernel", ylab = "K(x)")
```



Das Problem ist jetzt, dass wir einen eindensionalen Kernel verwenden wollen, um eine zweidimensionale Funktion (inputdimension) zu schätzen. Im Sinne der Einfachheit multiplizieren wir daher die Kernel auf jeder Dimension:

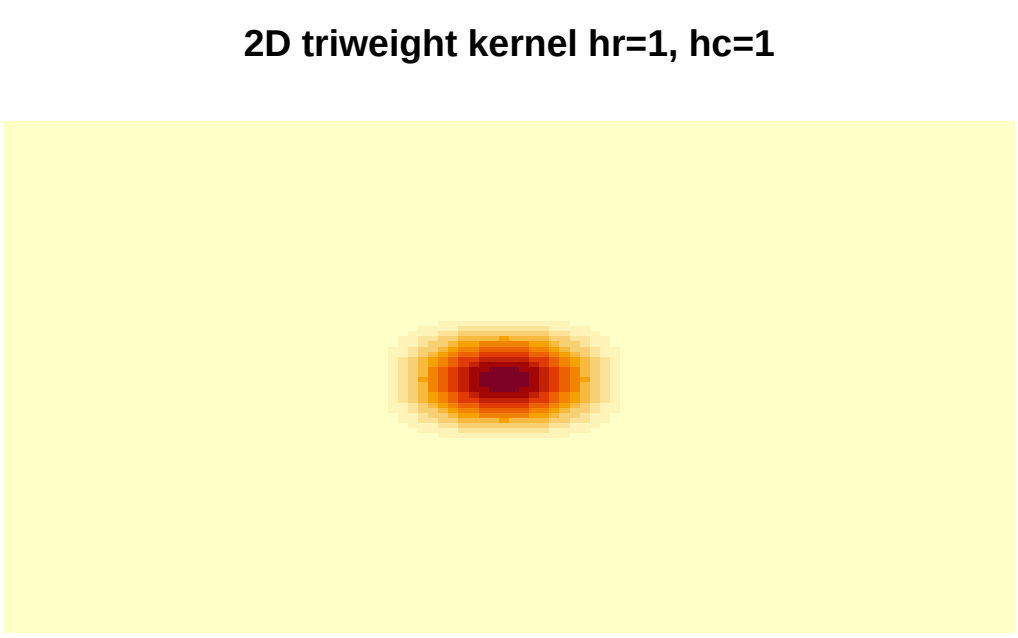
$$K_{h_r,h_c}(u,v) = K_{h_r}(u)K_{h_c}(v)$$

Das erleichtert auch die Berechnung der benötigten Gewichtungen. Es seien ω_N und ω_M die Fourier-Frequenzen nach Zeile und Spalte des räumlichen $N \times M$ Gitters. Die Gewichte für die Kerndichteschätzung lassen sich somit bestimmen aus:

$$
\begin{aligned}
K_{h_r,h_c}(\omega_N - \omega_1, \omega_M - \omega_2) &= \begin{pmatrix} K_{h_r,h_c}(\omega_{N1} - \omega_1, \omega_{M1} - \omega_2) & \cdots & K_{h_r,h_c}(\omega_{N1} - \omega_1, \omega_{MM} - \omega_2) \\ \vdots & \vdots & \vdots \\ K_{h_r,h_c}(\omega_{NN} - \omega_1, \omega_{M1} - \omega_2) & \cdots & K_{h_r,h_c}(\omega_{NN} - \omega_1, \omega_{MM} - \omega_2) \end{pmatrix} \\
&= \begin{pmatrix} K_{h_r}(\omega_{N1} - \omega_1)K_{h_c}(\omega_{M1} - \omega_2) & \cdots & K_{h_r}(\omega_{N1} - \omega_1)K_{h_c}(\omega_{MM} - \omega_2) \\ \vdots & \vdots & \vdots \\ K_{h_r}(\omega_{NN} - \omega_1)K_{h_c}(\omega_{M1} - \omega_2) & \cdots & K_{h_r}(\omega_{NN} - \omega_1)K_{h_c}(\omega_{MM} - \omega_2) \end{pmatrix} \\
&= \begin{pmatrix} K_{h_r}(\omega_{N1} - \omega_1) \\ \vdots \\ K_{h_r}(\omega_{NN} - \omega_1) \end{pmatrix} \begin{pmatrix} K_{h_c}(\omega_{M1} - \omega_1) & \cdots & K_{h_c}(\omega_{MM} - \omega_1) \end{pmatrix} \\
&= K_{h_r}(\omega_N - \omega_1)K_{h_c}(\omega_M - \omega_2)^T
\end{aligned}
$$

Die Kerndichteschätzung lässt sich jetzt schreiben als:

```
M <- 100
N <- 100

omega_M <- 2*pi*((-((M - 2) %/% 2)):(M %/% 2))/M
omega_N <- 2*pi*((-((N - 2) %/% 2)):(N %/% 2))/N
image(tr_k(omega_M, 1) %*% t(tr_k(omega_N, 1)), axes = F, main="2D triweight kernel hr=1, hc=1")
```



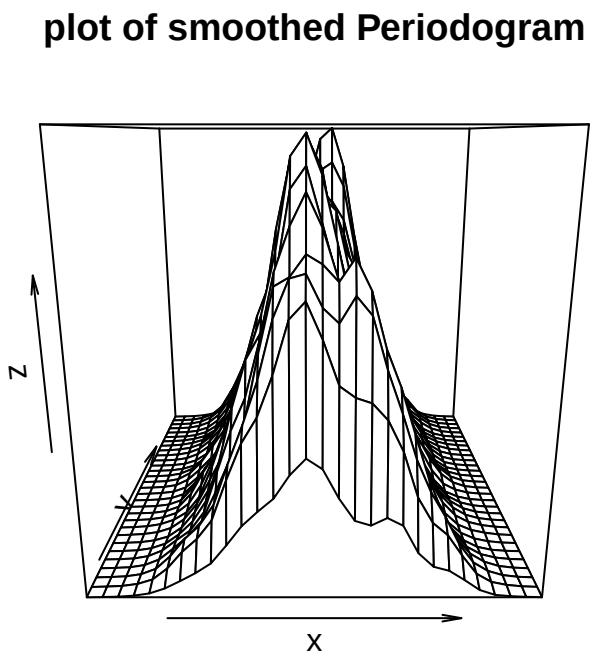
Das smooted periodogram lässt sich jetzt als Hadamard-Produkt der Gewichte und der Einträge des Peri-

odogramms darstellen:

$$\begin{aligned}\hat{f}(\omega_1, \omega_2) &= \frac{1}{NMh_r h_c} \sum_{k=1}^{N} \sum_{l=1}^{M} K_{h_r}(\omega_{Nk} - \omega_1) K_{h_c}(\omega_{Ml} - \omega_2) I(\omega_{Nk}, \omega_{Ml}) \\ &= \frac{1}{NMh_r h_c} \sum_{k=1}^{N} \sum_{l=1}^{M} \left[K_{h_r}(\omega_N - \omega_1) K_{h_c}(\omega_M - \omega_2)^T \odot I(\omega_N, \omega_M) \right]_{kl}\end{aligned}$$

Die Implementierung benutzt diese Darstellung. Darüber hinaus wird das Ergebniss des Periodogramms zwischengespeichert und bei jedem Funktionsaufruf wiederverwendent. Die Gewichte aus dem Kernel müssen dagegen leider bei jedem Funktionsaufruf neu berechnet werden. Das macht die Funktion insbesondere für größere Gitter teuer.

```
I_smooth <- function(I, kernel, hr = .5, hc = .5) {
  N <- nrow(I)
  M <- ncol(I)

  omega_M <- 2*pi*((-((M - 2) %/% 2)):(M %/% 2))/M
  omega_N <- 2*pi*((-((N - 2) %/% 2)):(N %/% 2))/N
  I_s <- function (omega_1, omega_2) {
    K_M <- kernel(omega_M - omega_1, hr)
    K_N <- kernel(omega_N - omega_2, hc)

    weights <- K_N %*% t(K_M)
    weighted <- sum(weights * I)

    normalized <- (1/(N*M*hr*hc))*weighted
    return(normalized)
  }
  class(I_s) <- "I_smooth"
  I_s
}

# implementing plot generic for quick and easy use
plot.I_smooth <- function(I_s) {
  x <- seq(-pi+1e-5, pi-1e-5, length.out = 30)
  y <- seq(-pi+1e-5, pi-1e-5, length.out = 30)


  z <- matrix(0, 30, 30)
  for (i in 1:length(x)) {
    for (j in 1:length(y)) {
      z[i, j] <- I_s(x[i], y[j])
    }
  }

  persp(x, y, z, main = "plot of smoothed Periodogram")
}

x <- gridMA(100, 100, K)
plot(I_smooth(I(x), tr_k))
```



T_n oder `Tn(x,y)`

Die Implementierung von T_n besteht aus zwei teilen. `Tn_f` ist die Funktion, welche am Ende auf der Fläche $[-\pi, \pi]^2$ integriert werden soll und `Tn` bildet das Integral. Für `Tn_f` wird einfach nur das Periodogramm von den beiden Pfaden gebildet und anschließend die Differenzen gebildet $I_x - \tilde{I}, I_y - \tilde{I}$. `Tn_f` ist dann das quadrierte smoothed periodogram über diese Differenzen. Für das Integrieren verwenden wir `int2` aus dem `rmutil` package.

```
# function to be integrated for Tn
Tn_f <- function(x,y, kernel, ...) {
  # cache periodogram and fourier frequencies
  I_x <- I(x)
  I_y <- I(y)
  I_tilde <- .5*(I_x + I_y)

  I_x_smooth <- I_smooth(I_x - I_tilde, kernel, ...)
  I_y_smooth <- I_smooth(I_y - I_tilde, kernel, ...)

  function(omega_1, omega_2) {
    return(
      I_x_smooth(omega_1, omega_2)^2 + I_y_smooth(omega_1, omega_2)^2
    )
  }
}
```

```
Tn <- function(x, y, kernel = tr_k, ...) {
  l2 <- Tn_f(x, y, kernel, ...)
  int2(l2, c(-pi, -pi), c(pi, pi))
}

# testing implementation under H_0
x <- gridMA(100, 100, K)
y <- gridMA(100, 100, K)

# Tn(x,y)
```

T_n^* oder `Tn_star(x,y)`

Der Aufbau von `Tn_star` ist sehr ähnlich zu `Tn`. In `Tn_star_f` geschiet der Tausch über Frequenzpaare durch ein Hadamard-Produkt.

```
Tn_star_f <- function(x, y, kernel = tr_k, ...) {
  I_x <- I(x)
  I_y <- I(y)

  I_tilde <- .5*(I_x + I_y)

  # permuting entries
  N <- nrow(I_x)
  M <- ncol(I_x)
  perm <- matrix(runif(N*M), ncol = N)
  perm <- as.numeric(perm > 0.5)
  perm_n <- as.numeric(!perm)
  I_x_rand <- I_x*perm + I_y*perm_n
  I_y_rand <- I_x*perm_n + I_y*perm

  I_x_diff <- I_smooth(I_x_rand - I_tilde, kernel, ...)
  I_y_diff <- I_smooth(I_y_rand - I_tilde, kernel, ...)

  function(omega_1, omega_2) {
    return(
      I_x_diff(omega_1, omega_2)^2 + I_y_diff(omega_1, omega_2)^2
    )
  }
}

Tn_star <- function(x, y, kernel = tr_k, ...) {
  l2_rand <- Tn_star_f(x, y, kernel, ...)
  int2(l2_rand, c(-pi, -pi), c(pi, pi))
}
#
# Tn_star(x, y)
```

Resampling Verfahren φ_n^*

Das Resampling ist sehr ähnlich zu den Schritten aus dem theoretischen Teil. Die Funktion gibt zudem auch ein paar zusätzliche Informationen aus. Unter anderem wird die Entwicklung von T über alle Bootstrap-

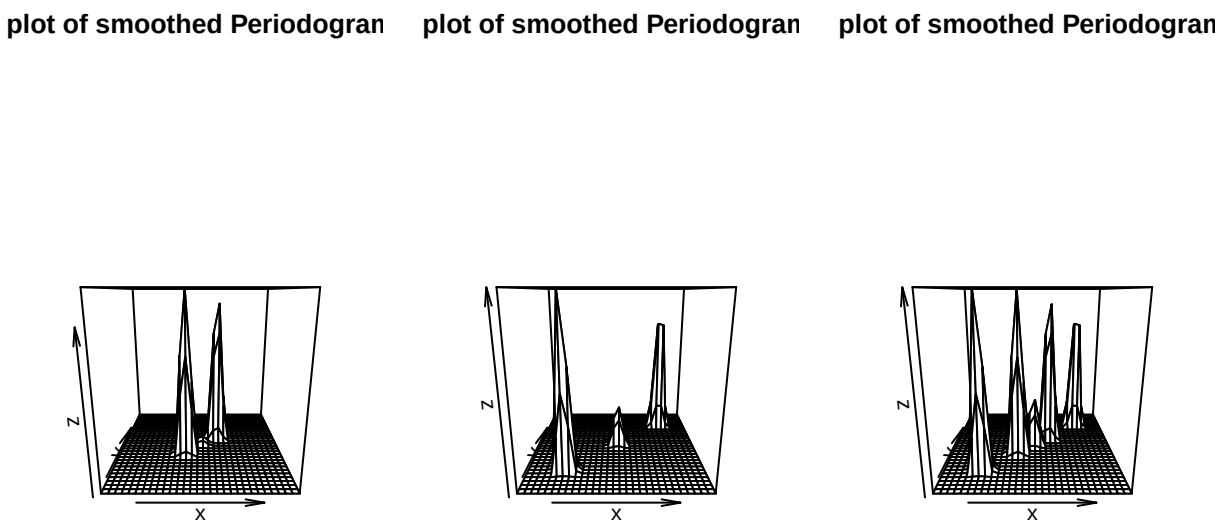
Ziehungen geplottet, um zu schauen, ob das Quantil bereits konvergiert.

```
phi_n <- function(x, y, B, alpha, kernel = tr_k, print_result = T, ...) {
  Tn_val <- Tn(x, y, kernel, ...)

  Tn_star_val <- numeric(B)
  T_val <- numeric(B)
  for (i in 1:B) {
    Tn_star_val[i] <- Tn_star(x, y, kernel, ...)
    T_val[i] <- sum(Tn_star_val[1:i] < Tn_val)/i
  }

  if (print_result == T) {
    par(mfrow = c(1,2))

    hist(Tn_star_val, main = "Histogram of Tn*",
         xlim = c(min(c(Tn_star_val, Tn_val)), max(c(Tn_star_val, Tn_val))))
    abline(v = Tn_val, col = "red")
    plot(T_val, main = "Trace of T over all bootstraps",
         xlab = "b",
         ylab = "T",
         ylim = c(0,1),
         type = "l")
    abline(h = alpha, col = "red")

    print("Test for equality of spatial ACF and spectral density")
    print(paste("Bootstrap samples: ", B))
    print(paste("Tn: ", Tn_val))
    print(paste("Percentage of Tn > Tn*: ", T_val[B]*100, "%"))
    if (T_val[B] > alpha) {
      print("Decision: Rejecting H0")
    } else {
      print("Decision: Accepting H0")
    }
  } else {
    return(alpha_hat)
  }
}

# testing for H0
# K <- matrix(.5, 3, 3)
# K[2,2] <- 1
# x <- gridMA(50, 50, K)
# y <- gridMA(50, 50, K)
#
# phi_n(x, y, 25, .05)
#
# # testing for H1
# x <- gridMA(50, 50, K)
# K_tilde <- matrix(.3, ncol = 3, nrow = 3)
# K_tilde[2,2] <- 1
# y <- gridMA(50, 50, K_tilde)
#
# phi_n(x, y, 25, .05)
```

Das Ergebnis dieser Tests lässt sich leider schwer vorhersagen, aber aus persönlicher Erfahrung scheint das Testverfahren relativ willkührlich. Gründe dafür könnten sein, dass die Spektraldichten bei dem gewählten Prozess sehr rauschen oder sich für K und $\tilde{K}$ nicht genug unterscheiden.

Daher verfolgen wir jetzt einen anderen Ansatz. Der folgende Prozess wird aus einem einzigen Signalpaar erzeugt. Dadurch erhalten nicht nur eine stabile Spektraldichte, sondern können auch Spaktraldichten erzeugen, welche sich sehr unterscheiden.

```
# Different approach signal based process:
signalProcess <- function(N, M, s_1 = 20, s_2 = 20){
  outer(
    1:N,
    1:M,
    FUN = function(i, j) {
      # signal created by
      sin(2*pi * (s_1*i+s_2*j)/(N+M)) + rnorm(1)*.3
    }
  )
}

I_inner <- I(signalProcess(100, 100))
I_outer <- I(signalProcess(100, 100, 60, 60))
I_inner_s <- I_smooth(I_inner, tr_k)
I_outer_s <- I_smooth(I_outer, tr_k)

I_tilde <- I_smooth(.5*(I_inner + I_outer), tr_k)

par(mfrow = c(1,3))
plot(I_inner_s)
plot(I_outer_s)
plot(I_tilde)
```



Nun wenden wir das Testverfahren auf Daten aus der Alternative und H_0 an:

```
x <- signalProcess(100, 100)
y <- signalProcess(100, 100)

# phi_n(x,y, 25, .05)
```

```
x <- signalProcess(100, 100, 60, 60)
y <- signalProcess(100, 100)

# phi_n(x,y, 25, .05)
```

Leider haben wir bei diesen Prozessen ebenfalls keinen Erfolg. Es deutet stark auf einen Fehler meinerseits hin, den ich aber leider nicht finden konnte.

Größere simulations studien

Außerhalb von kleineren Berechnungen der Teststatistik wurden auch Tests im größeren Umfang in der Cloud ausgeführt.

Literatur

Jentsch, Carsten, and Markus Pauly. 2015. "Testing Equality of Spectral Densities Using Randomization Techniques."