Fourier Analyse: MATLAB basierte Lehre und Anwendungen

Hans G. Feichtinger
WEBPAGE: www.nuhag.eu

München, May 10th, 2016
MATLAB EXPO 2016

Overview over my presentation

- Persönlicher Hintergrund
- Hintergrund und historischer Abriss
- MATLAB und Lineare Algebra
- Von der Linearen Algebra zur DFT/FFT
- Polynome und die DFT/FFT
- Anwendungen der FFT, TILS
- Unkonventionelle Anwendungen
- Zeit-Frequenz-Analyse und Gabor Analyse
- Links und Internet Resourcen
- Zusammenfassung

Kurzer Lebenslauf von Hans G. Feichtinger

- Lehramt Mathematik und Physik, Univ. Wien;
- Habilitation im Fach Mathematik 1979 in Wien;
- bis zur Pensionierung (Dez. 2015) an der Univ. Wien;
- diverse Gastprofessuren weltweit;
- seit 1992 Aufbau der Numerischen Harmonischen Analyse Gruppe NuHAG (www.nuhag.eu) in Wien;
- seit 1979 habilitiert (Univ. Wien, Mathematik), 27 PhD students laut Mathematical Genealogy.
- ab 2000 Chief Editor: J. Fourier Analysis & Applications;
- zahlreiche theoretische und angewandte Projekte.

HGFei: Forschungsprofil

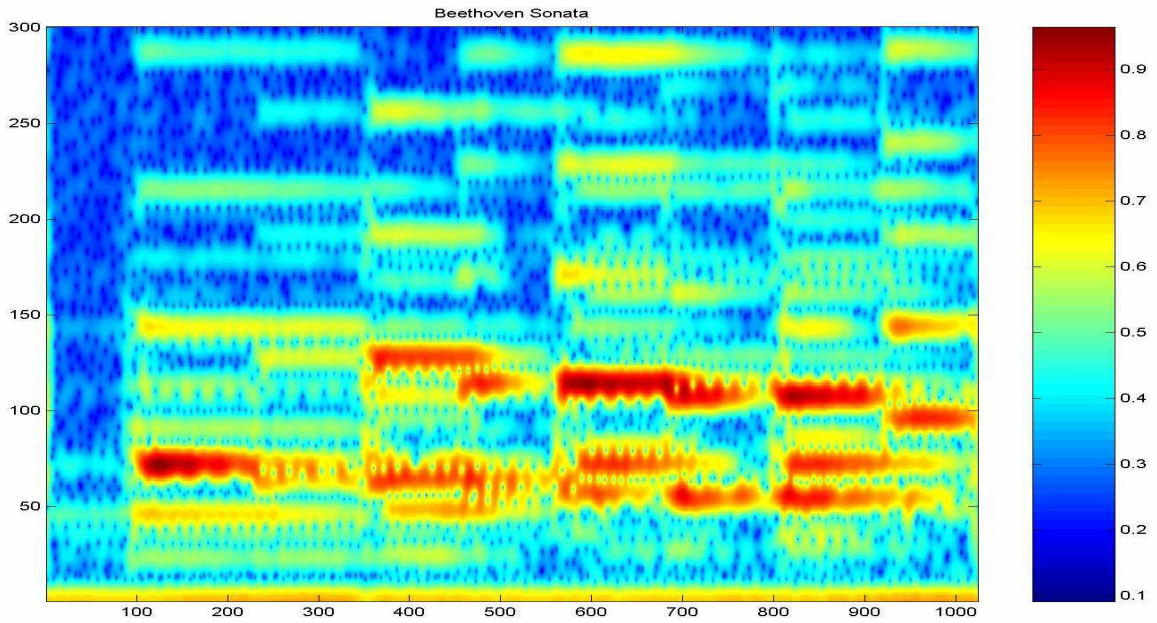
Ich bezeichne mich als **anwendungsorientierten Mathematiker**, mit starkem Interesse für die mathematischen Grundlagen der Signal- und Bildverarbeitung (die als WAV-files, JPEG-Bildern, MP3-Dateien zum täglichen Leben gehören).

Genauer, sehe ich mich als **anwendungsorientierter Vertreter der Numerischen Harmonischen Analyse**, der Konzepte und Algorithmen entwickelt und propagiert.

Mein besonderes Interesse gilt dem Begriff der **Conceptual Harmonic Analysis**, die Distributionentheorie aber auch Numerische Fourier Analysis umfaßt. Gabor Analysis und Funktionenräume gehören zu meinen Spezialgebieten.

Gabor Analysis: Beethoven Piano Sonata

Gabor Analysis ist die mathematische Theorie des Spektrograms: Welche Fenster, welche mögliche Diskretisierung, etc.. Zur Illustration: das Spektrogramm einer Beethoven Klavier Sonate!



Wo und warum ist die Fourier Transformation wichtig?

Die Fourier Transformation (FT), insbesondere die Diskrete Fourier Transformation (DFT) in der Implementierung als FFT (Fast Fourier Transform) ist aus unserem digitalen Zeitalter nicht mehr wegzudenken. Die meisten Algorithmen der digitalen Signal- oder Bildverarbeitung basieren auf der FFT bzw. der FFT2.
Mit den folgenden Begriffen sind Studierende der Ingenieurswissenschaften oft schon früh konfrontiert:

1. lineare translationsinvariante Systeme (LTIS);
2. Impuls-Antwort und Transfer-Funktion;
3. Faltung, Fourier Transformation,
4. Sampling und Shannon's Abtasttheorem,

um nur die wichtigsten zu nennen.

Schul-Multiplikation und Systemtheorie

Wir wollen eine konkrete Form von "translations-invarianten Systemen" an den Anfang stellen, die wohl jede/r von der Schule her kennt: Das MULTIPLIZIEREN mit einer natürlichen Zahl, hier mit 123. Betrachten wir die folgende Aufgabe und ihre naheliegende Umsetzung durch einen smarten Schüler, der nicht einfach dem allgemeinen Rezept stur folgt, so ist folgender Rechengang naheliegend

```
450002 x 123 = 450000 x 123 + 2 x 123
= (45 x 123) x 10000 + 246
= 55350000 + 246 = 55350246
```

Verwendet offenbar additives splitting $(450000 + 2)$ und Verschieben der Kommastelle (nachher/vorher). Natürlich genügt es, den Wert `1 x 123 = 123` zu kennen, um zu wissen, wie ein beliebiges $n \in \mathbb{N}$ multipliziert wird!

Multiplikation und Faltung

Eine ähnliche Multiplikationsübung ergibt

$$1111110^2 = 1234565432100$$

Man kann aber auch leicht sehen, dass die Struktur des Ausmultiplizieren praktisch dieselbe ist, wie die Bestimmung der Koeffizienten des Polynoms $p(x)^2$, wenn $p(x)$ gegeben ist als:

$$p(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x,$$

und zwar haben wir konkret

$$p(x)^2 = x^{12} + 2x^{11} + 3x^{10} \cdots + 4x^5 + 3x^4 + 2x^3 + x^2.$$

Die Koeffizienten dieses Polynoms haben auch etwas mit der Wahrscheinlichkeit der Summe von zwei Würfeln (in 1/36 ausgedrückt), zu tun!

Das klassische Fourier Programm

1. J. B. Fourier: 1822: *Jede periodische Funktion hat eine Darstellung als eine unendliche Summe von "harmonischen Schwingungen"*: $f(t) = \sum_{k\in\mathbb{Z}} c_k e^{2\pi ikt}$ (Fourier Reihe);
2. Die Klärung der Konvergenzfrage >> Lebesgue Integral;
3. zufriedenstellende Theorie f.d. Hilbertraum $\left(\boldsymbol{L}^2(\mathbb{U}), \|\cdot\|_2\right)$;
4. Übergang zur unendlichen Periode (kontuierliches Frequenzspektrum) >> *Fourier Transformation*;
5. Fouriertransformation und Umkehrformel für $\boldsymbol{L}^1(\mathbb{R}^d)$; Satz von Plancherel: $\|f\|_2 = \|\hat{f}\|_2$;
6. L. Schwartz führte FT f. temperierte Distributionen ein;
7. DFT = Discrete Fourier Transf., FFT Algorithmus von Cooley-Tuckey (1965).

Definition der Diskreten Fourier Tranformation (DFT)

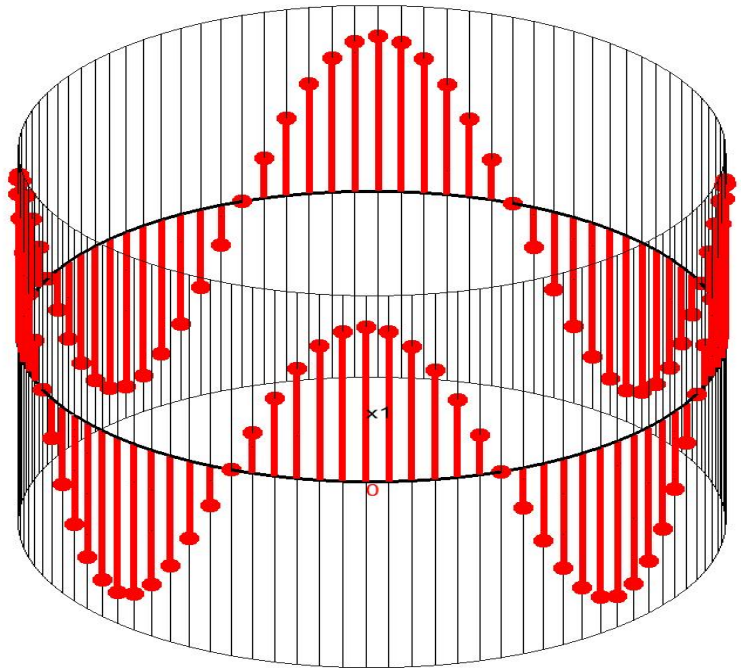
Wir schreiben $f = [f(0), \cdots, f(L-1)] \in \mathbb{C}^L$ (komplexe Vektoren der Länge L), und betrachten f als **periodisches, diskretes Signal** (mit Periode L), mit $i = \sqrt{-1}$ wie üblich:

$$\widehat{f}(k) = \sum_{l=0}^{L-1} f(l) e^{-\frac{2\pi i k \cdot l}{L}} \qquad k = 0, \cdots, L-1 \qquad (1)$$

$$f(n) = \frac{1}{L} \sum_{l=0}^{L-1} \widehat{f}(l) e^{\frac{2\pi i n \cdot l}{L}} \qquad n = 0, \cdots, L-1 \qquad (2)$$

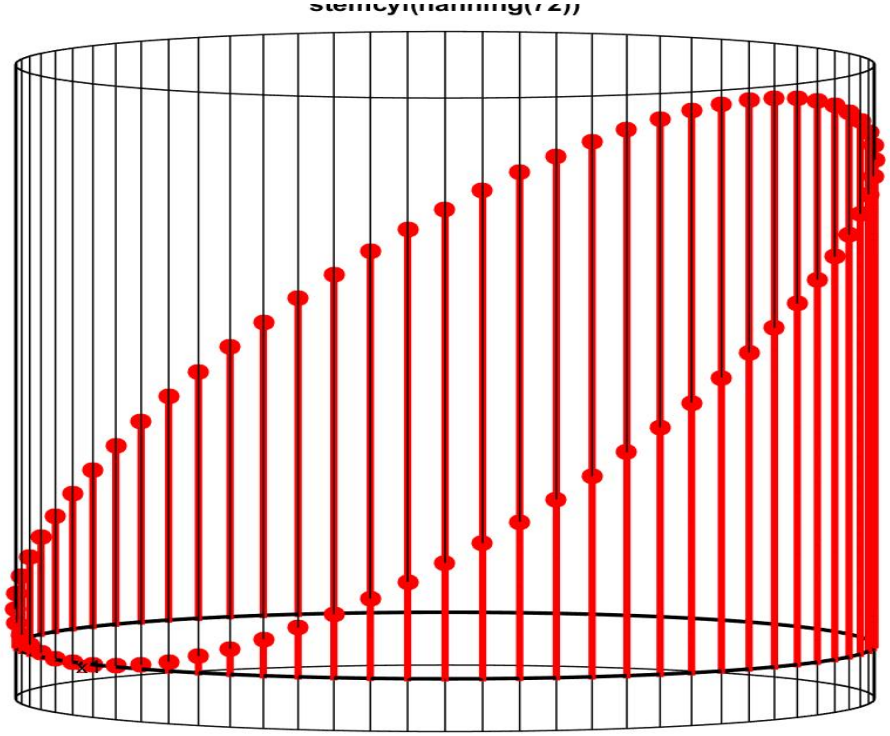
Illustration eines Hanning Fensters

Darstellung einer periodischen Folge mittels `STEMCYL` (NuHAG M-file). Ist auch Grund für die Verwendung der zyklischen Verschiebung (Rotation!).



Was bedeutet die Indizierung (nullte Komponente?)

Nebenbemerkung: Diese Art der Darstellung, die auch Rotation des Objectes erlaubt und Kippen, erlaubt auch interessante Demonstrationen, beispielsweise hat ein *Hanning-Fenster* eine einfache Darstellung: (`hanning(72)` wird dargestellt:



Die DFT in MATLAB Beschreibung

Da Vektoren keine "nullte Komponente haben", ist es besser (wenn man die DFT naiv aufgrund der Formel in MATLAB realisieren will) folgende Konvention zu verwenden: Für $\mathbf{x} = [x[1], \cdots, x[L]] \in \mathbb{C}^L$ ist die DFT (Discrete Fourier Transform) $\mathbf{y} = DFT(\mathbf{x})$ definiert als:

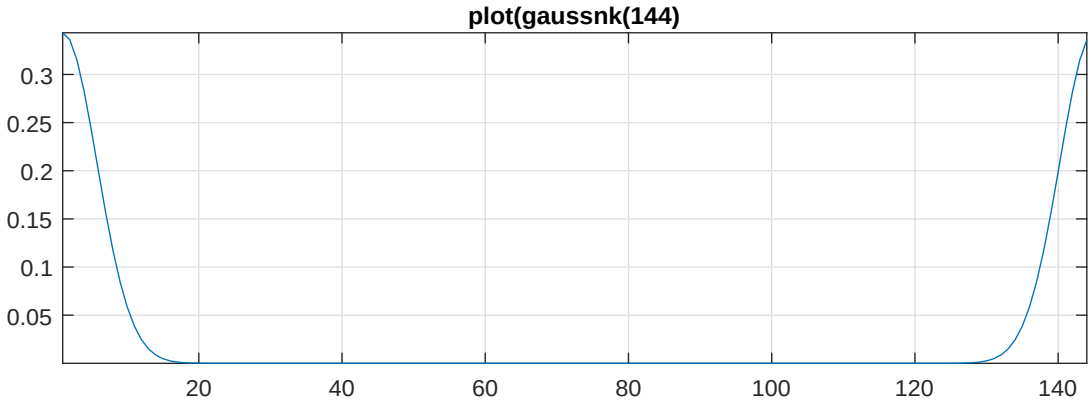
$$y[k] = \sum_{l=1}^{L} x[l] e^{-\frac{2\pi i (k-1)\cdot(l-1)}{L}} \qquad k = 1, \cdots, L. \tag{3}$$

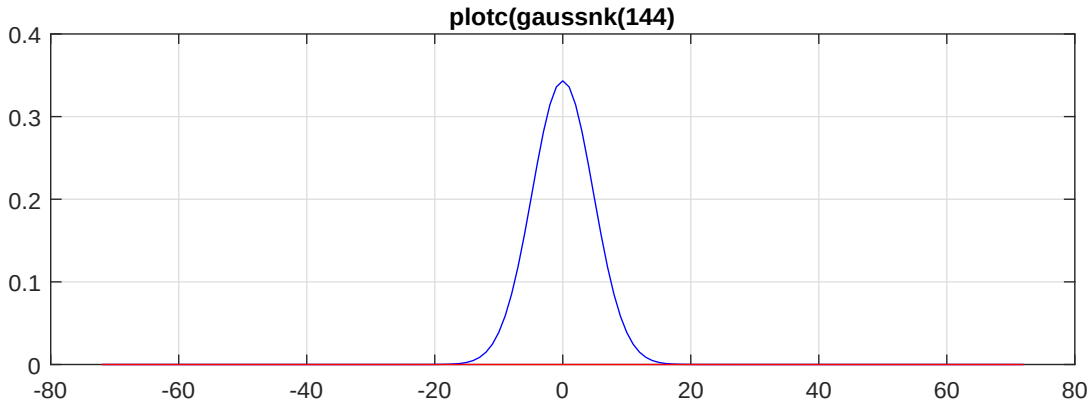
$$x[n] = \frac{1}{L} \sum_{k=1}^{L} y[l] e^{\frac{2\pi i (n-1)\cdot(k-1)}{L}} \qquad n = 1, \cdots, L. \tag{4}$$

VORTEIL: Korrekte Indizierung, aber schwierige Interpretation der Koeffizienten! Außerdem gibt es ein Plotproblem!

Korrektes Ploten einer zentrierten Gaussfunktion

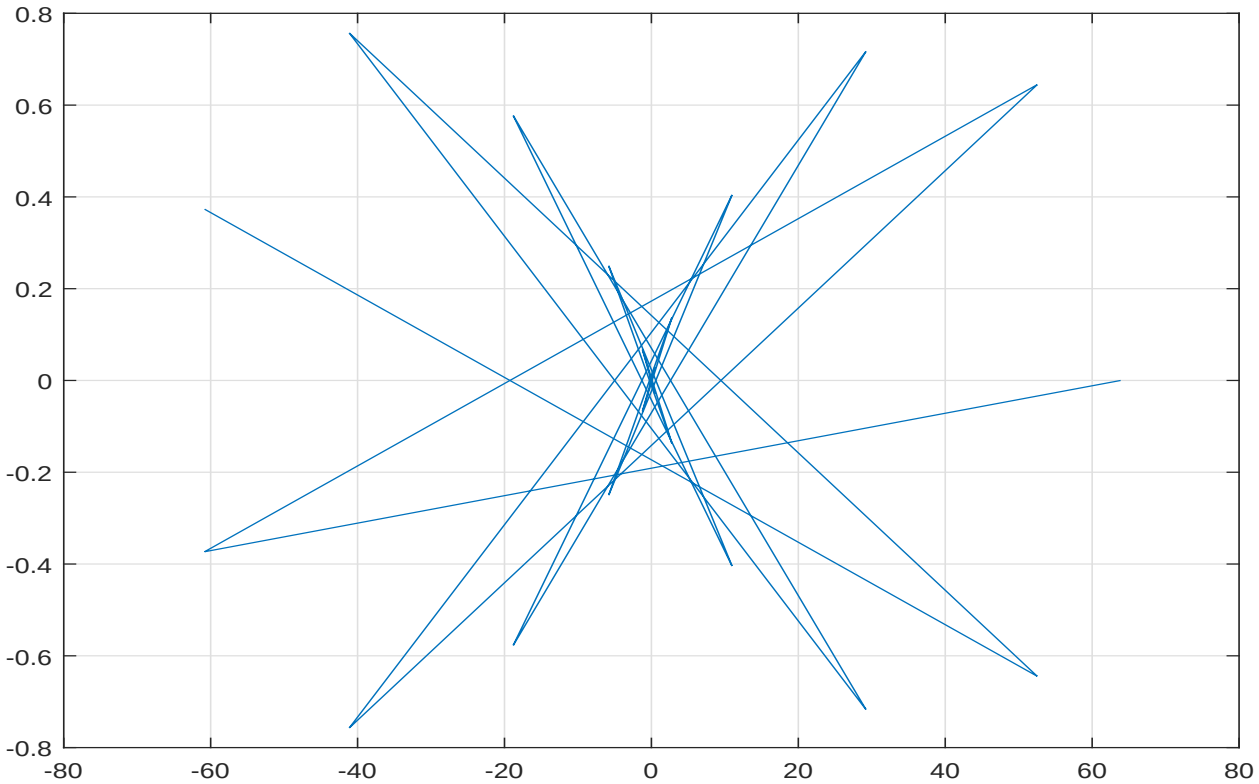
Die “richtige” Datenstruktur (diese diskrete Gauss-Funktion ist FFT invariant) und deren “korrekte” Plot-Darstellung (PLOTC.M).





Diskrete, Fourier-invariante Gauss-Funktion

Die naive Befehlsfolge kann nur Verwirrung stiften: $bas = linspace(-4, 4, 512); gdisc = exp(-pi * bas.^2); plot(fft(gdisc))$



Alternativen zum klassischen Programm?

Ich behaupte, dass **MATLAB** eine Möglichkeit bietet, *von der Seite der linearen Algebra* den Einstieg in die wesentlichen Aspekte der Fourier Analysis auf einem anwendungsorientierten, aber ebenso mathematisch korrekten Weg klarzumachen.

So wie die reellen Zahlen als Grenzwerte von einfachen rationalen Zahlen beschrieben werden können, sollte man die kontinuierliche Theorie nur als Grenzfall von diskreten Signalen gesehen werden (so wie Pixel Bilder eine gute Approximation von kontinuierlichen Bildern sind!).

Fangen wir mit **Fakten aus der Linearen Algebra** (für komplex-wertige Vektoren und Matrizen/Vektoren) an!

Linearen Algebra und MATLAB I

Erinnern wir uns an die Grundbegriffe der Linearen Algebra und ihre Realisierung mittels MATLAB, d.h. die Rolle von Matrizen und Matrix-Multiplikation:

Definition

Ein Vektor $\mathbf{b}$ ist eine Linearkombination, d.h. von der Form $\mathbf{b} = x_1\mathbf{a}_1 + \cdots x_n\mathbf{a}_n$ ist gleichbedeutend mit $\mathbf{b} = \boldsymbol{A} * \mathbf{x}$, wobei $(\mathbf{a}_k)_{k=1}^n$ die Spalten von $\boldsymbol{A}$ in $\mathbb{C}^m$ (oder $\mathbb{R}^m$) sind.

Dementsprechend sind Linear-Kombinationen von Linear-Kombinationen wieder Linear Kombinationen (der ursprünglichen Vektoren), und $\boldsymbol{B} = \boldsymbol{A}_2 * \boldsymbol{A}_1$ sagt uns, welche Koeffizienten man braucht. Die eindeutigen Koeffizienten für $\mathbf{b}$ bekommt man mittels $\mathbf{x} = \boldsymbol{A}^{-1} * \mathbf{b}$.

Linearen Algebra und MATLAB II

Definition

Eine Abbildung T von $\mathbb{C}^n$ nach $\mathbb{C}^m$ heisst *linear*, wenn sie Linearkombinationen respektiert, d.h. wenn gilt

$$T\left(\sum_k c_k \mathbf{b}_k\right) = \sum_k c_k T(\mathbf{b}_k).$$

Konsequenterweise kennt man eine linear Abbildung, sobald man sie nur auf den Basis Elementen kennt, und es ist naheliegend, T durch eine Matrix zu beschreiben, die einfach die (Koordinaten) der Bilder der Basisvektoren enthält, also durch eine $m \times n$ Matrix, mit $\mathbf{a}_k = T(\mathbf{e}_k), 1 \leq k \leq n$.

Verträglich mit Matrix Komposition und Inversion!

Linearen Algebra und MATLAB III

Es gibt in MATLAB keinen eigenen Befehl für das Skalarprodukt (in $\mathbb{C}^n$), weil $\langle \mathbf{x}, \mathbf{y} \rangle$ ohnehin durch den Befehl $\mathbf{y}' * \mathbf{x}$ realisiert werden kann. Die bekannte Formel

$$\langle \boldsymbol{A} * \mathbf{x}, \mathbf{y} \rangle = \langle \mathbf{x}, \boldsymbol{A}' * \mathbf{y} \rangle$$

ergibt sich so aus der Assoziativität der Matrix-Multiplikation und der bekannten Regel

$$(\boldsymbol{A} * \boldsymbol{B})' = \boldsymbol{B}' * \boldsymbol{A}'.$$

Insbesondere ist $\mathbf{y} \mapsto \boldsymbol{A}' * \mathbf{y}$ für $\mathbf{y} \in \mathbb{C}^m$ gleichbedeutend mit

$$\mathbf{y} \mapsto \left(\langle \mathbf{y}, \mathbf{a}_k \rangle \right)_{k=1}^{n}.$$

Linearen Algebra und MATLAB IV

Warum sind unitäre Matrizen so nützlich?
Daraus folgt wieder unmittelbar (mit ein wenig Theorie betreffend inverse Elemente) dass eine $n \times n$-Matrix $\mathbf{U}$, deren Spalten ein Orthonormalsystem bilden, d.h. mit $\mathbf{U}' * \mathbf{U} = \mathrm{Id}_\mathbf{n}$, auch

$$\mathbf{U} * \mathbf{U}' * \mathbf{x} = \mathbf{U} * (\mathbf{U}' * \mathbf{x}) = \mathbf{x}, \quad \mathbf{x} \in \mathbb{C}^n$$

erfüllen (bzw. $\mathbf{U}^{-1} = \mathbf{U}'$!), d.h. es gilt für $\mathbf{x} \in \mathbb{C}^n$:

$$\mathbf{x} = \sum_{k=1}^{n} \langle \mathbf{x}, \mathbf{u}_k \rangle \mathbf{u}_k,$$

m.a.W.: die eindeutigen Koeffizienten bzgl. der Basis $(\mathbf{u}_k)$ sind durch die Skalarprodukte mit diesen Vektoren gegeben!

Linearen Algebra und MATLAB V

Beispiel: Die Abbildung, welche den Koeffizienten $\vec{a}$ eines Polynoms (entsprechend den MATLAB Konventionen) mittels `polyval(a,z)` die Werte an den Stellen z_k in $\mathbb{C}$ zuordnet, also

$$p(z) = a_1 z^{n-1} + a_2 z^{n-2} \dots a_n.$$

Diese ist genau dann invertierbar, wenn wir n verschiedene Stellen betrachten, wenn wir also Daten $\vec{d} \in \mathbb{C}^n$ haben.

Dann gilt für die Vandermonde Matrix `vander(z)` mit $\vec{z} \in \mathbb{C}^n$

$$\vec{d} = polyval(a, z) = vander(z) * a$$

bzw.

$$a = inv(vander(z)) * d.$$

Von der Linearen Algebra zur Fourier Transformation

Da sich die DFT (die Diskrete FT), implementiert in Form der FFT (Fast FT) durch Lineare Algebra und folglich durch Matrizen-Rechnung verständlich gemacht werden kann, schlage ich einen Zugang von dieser Seite, unter Zuhilfe von MATLAB vor: Dazu kann man von der **FFT-Routine** ausgehen, und **experimentell einige Grundfakten verifizieren**.

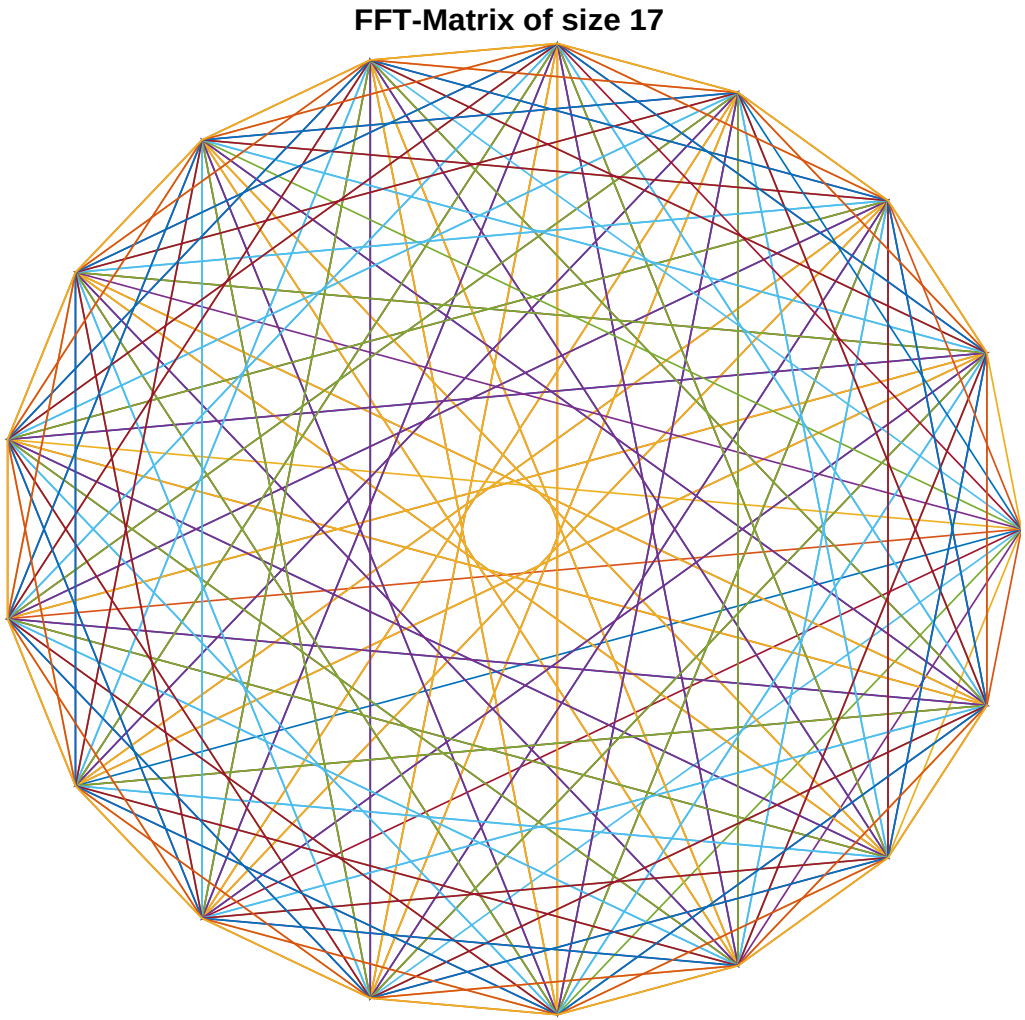
Da der *output* der FFT-Routine offensichtlich eine komplexwertige Folge ist, wollen wir die FFT gleich als eine Abbildung von $\mathbb{C}^N$ nach $\mathbb{C}^N$ ansehen (offensichtlich ist das output Format gleich dem input Format!), wie ein einfacher TEST zeigt:

```
fft(rand(1,5)), fft(rand(4,1)), fft(rand(3,4)),
```

Von der Linearen Algebra zur Fourier Transformation II

1. **Die FFT bewahrt die (euklidische) Norm von Vektoren**, bis auf einen Faktor $\sqrt{N}$, der von der Länge abhängt.
   TEST: `x = rand(N,1);`
   `norm(fft(x))/(sqrt(N)*norm(x))`,
2. **die FFT ist linear**, wie durch folgenden Test "verifizert" wird: `A = rand(3,4); x = rand(4,1);`
   `norm(fft(A)*x - fft(A*x))`,
   Nach den Konventionen **agiert die FFT auf einer Matrix spaltenweise**. Also zeigt der obige Test, dass die FFT-Routine eine zufällige Linear-Kombinationen respektiert!
3. `F = fft(eye(4)), norm(fft(x) - F * x)`
   zeigt, dass die Matrix Multiplikation mit F tatsächlich der Anwendung der FFT Routine entspricht.

Illustration der FFT-Matrix der Größe 17



Die DFT-Matrix als Vandermonde Matrix

Das vorhergehende Bild suggeriert schon, dass die N-ten Einheitswurzeln eine besondere Rolle spielen könnten, und tatsächlich kann man mit Hilfe von MATLAB leicht verifizieren, dass die Abbildung $\mathbf{x} \mapsto \mathrm{fft}(\mathbf{x})$ als Auswertungs-Abbildung eines Polynoms mit den Koeffizienten an den N-ten Einheitswurzeln interpretiert werden kann!

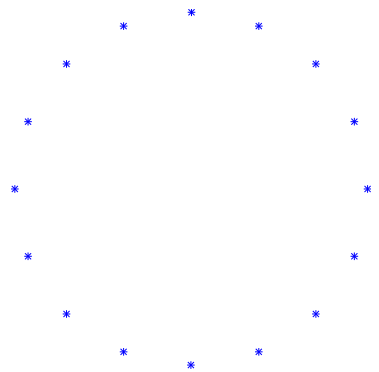
Es erhebt sich nur die Frage, nach welcher Konvention das Polynom (vom Grad $L-1$) aus der Koeffizientenfolge $\vec{a}$ (von Länge N) erzeugt wird, und in welcher Reihenfolge die N-ten Einheits- wurzeln anzugeben sind, um exakte (numerische) Übereinstimmung zu erzielen.

Die DFT-Matrix als Vandermonde Matrix II

Diese Aussage ist leicht auf folgende Art und Weise mittels MATLAB verifizierbar:

```
N = 16; bas = 0 : 1/N : (N-1)/N;
u = exp(2 * pi * i * bas);
plot(u,'b*'); axis square; axis off; shg;
norm( polyval(fliplr(aa),conj(u)) - fft(aa))
= approx. Null!
```

For illustration:



Die DFT-Matrix als Vandermonde Matrix III

Dabei ist zu beachten: Die Vandermonde Routine ist auf die MATLAB Konvention zum Befehl polyval abgestimmt, d.h. die Monome werden dort in absteigender Ordnung angeordnet, im Gegensatz zur traditionellen math. Beschreibung von Polynomen!

Entsprechend kann man auch die zugehörigen Matrizen identifizieren:

```
>> u5 = exp(2 * pi * i * (0:1/5:4/5));
>> V5 = vander(u5); V5g = V5(:,[5,1:4]);
>> norm(V5g - (fft(eye(5))) )
ans = 1.9239e-15
```

Also: Die FFT-Matrix entspricht der Vandermonde Matrix der Einheitswurzeln, beginnend by $1 = \omega_N^0$, im mathematisch positiven Sinne durchlaufen.

Die DFT-Matrix als Vandermonde Matrix IV

Alternativ betrachten wir die Vandermonde Matrix für $N = 5$, wobei die Einheitswurzeln im Uhrzeigersinn angeordnet seien:

```
>> u5 = exp(-2 * pi * i * (0:1/5:4/5));
```

Wegen der MATLAB Konventionen betreffend polyval adaptiert (mittels fliplr angepasst) ergibt:

```
>> VC5 = vander(u5); F5 = fft(eye(5));
>> norm(VC5 - fliplr(F5)
ans = 1.9239e-15
```

Also: Nach umgekehrter Anordnung der Spalten entspricht die FFT-Matrix der Vandermonde Matrix der Einheitswurzeln, beginnend by $1 = \omega_N^0$, im Uhrzeigersinn durchlaufen.

Von der Linearen Algebra zur Fourier Transformation III

Da nun $x \mapsto fft(x)$ eine lineare Abbildung ist, ist es interessant, die zugehörige Matrix weiter zu analysieren:

```
N = 17; F17 = fft(eye(N)); plot(F17); axis square;
```

Diese Matrix ist auch (reell) symmetrisch:

```
>> norm(F - F.', 'fro'), ans = 1.4316e-14
```

Andererseits ist es interessant, sich die Eintragungen dieser Matrix spaltenweise anzusehen. Etwa für $N = 144$:

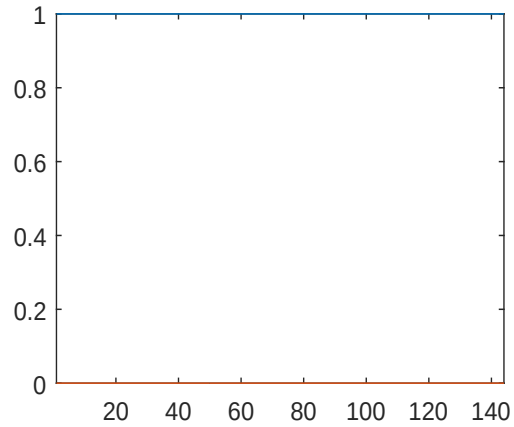
```
>> N = 144; F = fft(eye(N));
>> for jj=1:6; plot(1:N,real(F(:,jj)),1:N,
imag(F(:,jj))); shg; pause; end;
```

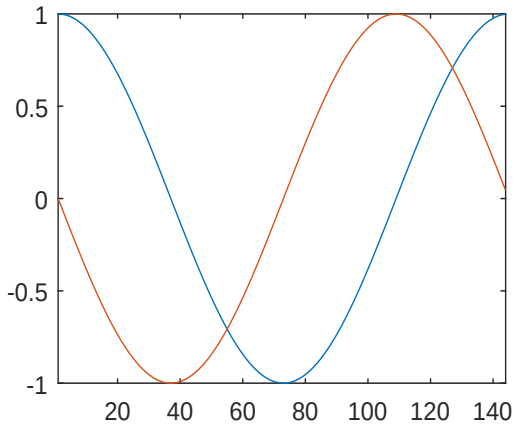
Die Spalten dieser Matrix haben alle gleiche Länge (: $\sqrt{N}$)

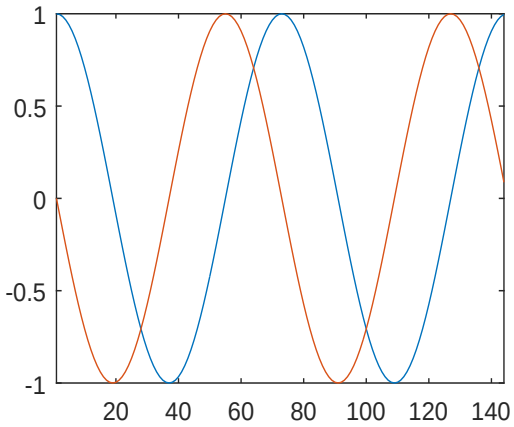
```
for jj=1:N; lF(jj)=norm(F(:,jj)); end;
norm(ones(1,N)*sqrt(N) - lF),
```

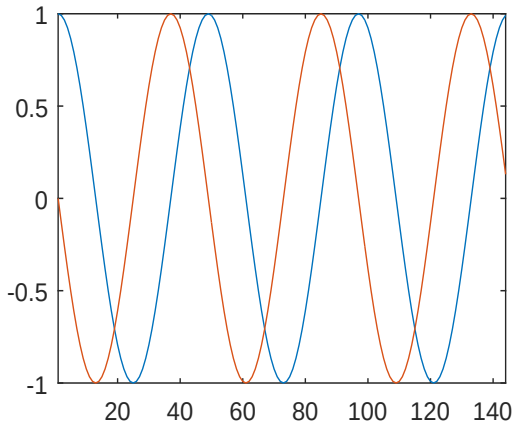
Die ersten Spalten der Fourier Matrix

Die Spalten der Fourier Matrix: die reinen Frequenzen!









Reine Frequenzen und Fourier Synthese

Die Spalten (= Zeilen) der Fourier Matrix sind die soeben gezeigten "reinen Frequenzen".
Nach der **Eulerschen Formel** sind sie von der Form

$$\chi_k(t) := e^{2\pi jkt} = \cos(2\pi kt) + j\sin(2\pi kt).$$

Diese sind "natürliche Bausteine", weil sie **Eigenvektoren der Translationsoperatoren** sind (mit komplexen! Eigenwerten), und zwar als Folge des Exponentialgesetzes:

$$[T_x\chi_k](t) := \chi_k(t-x) = e^{2\pi jk(t-x)} = e^{-2\pi jkx} \cdot \chi_k(t),$$

oder in Worten: Auf den sog. Charakteren χ_k wirkt der Verschiebungsoperator T_x einfach wie die Multiplikation mit der komplexen Zahl $e^{-2\pi jkx}$.

Orthogonalität der Spalten der Fourier Matrix

Natürlich kann man sich erhoffen, dass die FFT-Routine einem **orthogonalen Basis-Wechsel** entspricht. Zur Erinnerung:

```
A = rand(N); U = orth(A);
norm(U'*U - eye(N)), norm(U*U' - eye(N)),
```

Da die Spalten von F (und daher auch die Spalten von $F.' = F$) alle Länge $\sqrt{N}$ haben, kann man nur erwarten, dass **bis auf den Normierungsfaktor N die Matrix F' die inverse zu F ist**:

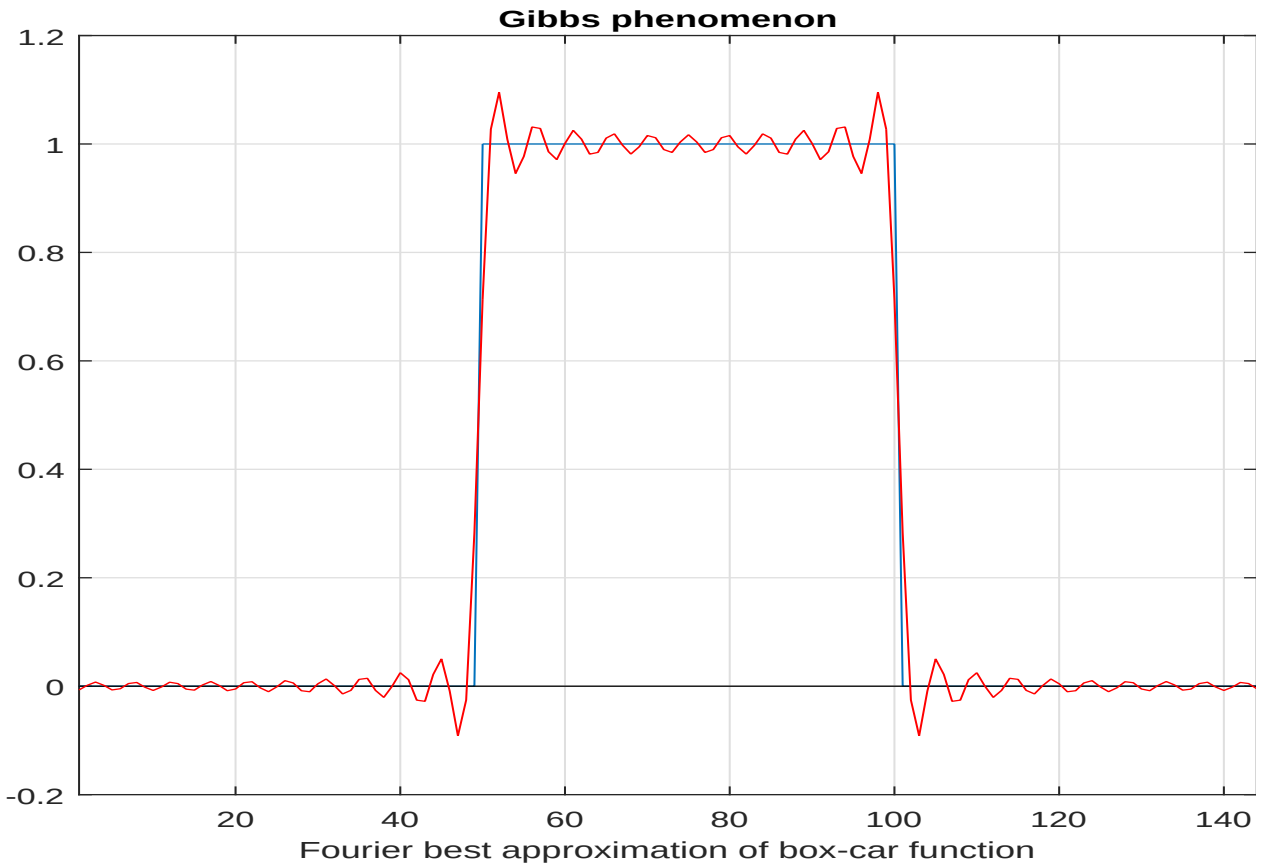
```
norm(F*F'/N - eye(N))
```

Die Bestapproximation durch gewisse Frequenzen, wobei J eine Teilmenge von $1 : N$ sei:

```
bx = zeros(1,N); bx(50:100) = 1; plot(bx);
ax20; grid; plotax;
```

Fourier Approximation of Box-car function

Die Best-Approximation (im Euklid. Sinne) der Rechtecksfunktion durch 61 Fourier-Koeffizienten (max. Frequenz 30):



Reine Frequenzen und Fourier Synthese

$$\mathbf{x} = F' * (F * \mathbf{x}/N), \quad \text{with} \quad F = fft(eye(N)),$$

zeigt also, dass jedes Signal als Summe der reinen Frequenzen geschrieben wird, und dass (bis auf den Faktor $1/N$) die Fourier Koeffizienten gerade die eindeutigen Koeffizienten sind.
Man spricht also bei der Darstellungsformel

$$\mathbf{x} = \sum_{k=0}^{N-1} c_k \chi_k$$

von der **Fourier Synthese**, und die Koeffizienten sind gerade von der Form $\mathbf{c} = fft(\mathbf{x})/N$. Dementsprechend ist $\mathbf{x} \mapsto fft(\mathbf{x})$ die **Fourier Analyse**.

Relevanz für die Systemtheorie

Einige der wichtigen Eigenschaften der Fourier Transformation ergeben sich daraus, dass sie die *Faltung* in punktweise Multiplikation überführt, und so dazu dient, *translations-invariante lineare Systeme* als Multiplikations-Operatoren darzustellen (*Transfer Funktion*).
Dazu beobachten wir die Wirkung einer *zyklischen Rotation*, in Matrix Format ($N = 7$, shift um 2 samples):

$$S2 := \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (5)$$

Relevanz für die Systemtheorie II

Konjugiert man die Standard-Matrix (an jeder Position steht die Koordinaten-Nummer, von 1 bis 49) mit $S2$, d.h. bildet man $\text{inv}(S2) * T7 * S2$, so bekommt man:

$$\begin{pmatrix} 41 & 48 & 6 & 13 & 20 & 27 & 34 \\ 42 & 49 & 7 & 14 & 21 & 28 & 35 \\ 36 & 43 & 1 & 8 & 15 & 22 & 29 \\ 37 & 44 & 2 & 9 & 16 & 23 & 30 \\ 38 & 45 & 3 & 10 & 17 & 24 & 31 \\ 39 & 46 & 4 & 11 & 18 & 25 & 32 \\ 40 & 47 & 5 & 12 & 19 & 26 & 33 \end{pmatrix} \tag{6}$$

d.h. alle Eintragungen wandern parallel zur Hauptdiagonale um zwei Einheiten nach rechts/unten.

Relevanz für die Systemtheorie III

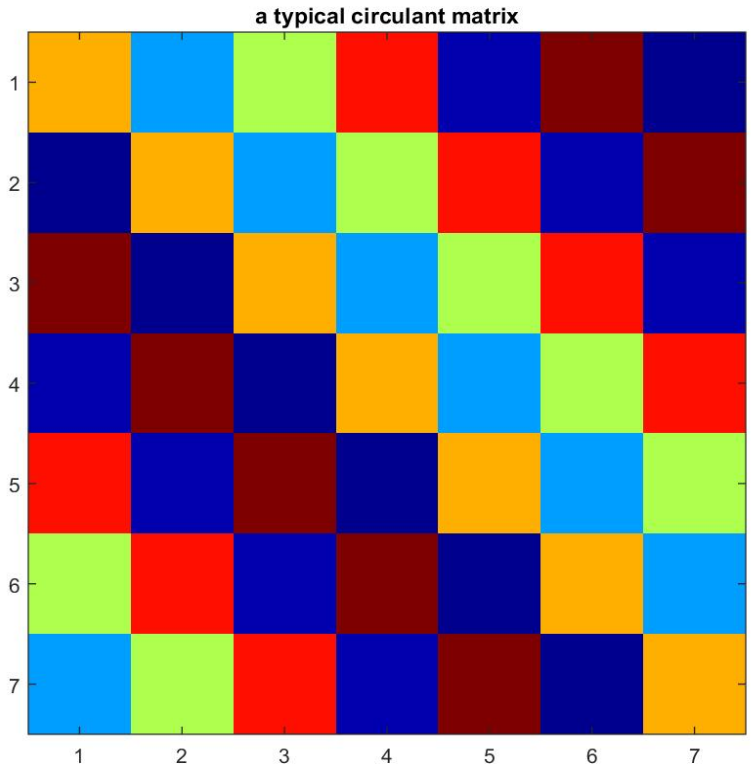
Entsprechend kommutiert eine 7×7 Matrix mit allen (zyklischen) Verschiebungen genau dann, wenn sie (im zyklischen Sinne) konstant auf den (zyklischen) Nebendiagonalen ist.
Wir verwenden dazu einen eigenen [NuHAG] Befehl:
`c = rand(1,7); CONVC = convmat(c),`

$$\begin{pmatrix} 0.7133 & 0.5072 & 0.6387 & 0.7903 & 0.3971 & 0.8663 & 0.3760 \\ 0.3760 & 0.7133 & 0.5072 & 0.6387 & 0.7903 & 0.3971 & 0.8663 \\ 0.8663 & 0.3760 & 0.7133 & 0.5072 & 0.6387 & 0.7903 & 0.3971 \\ 0.3971 & 0.8663 & 0.3760 & 0.7133 & 0.5072 & 0.6387 & 0.7903 \\ 0.7903 & 0.3971 & 0.8663 & 0.3760 & 0.7133 & 0.5072 & 0.6387 \\ 0.6387 & 0.7903 & 0.3971 & 0.8663 & 0.3760 & 0.7133 & 0.5072 \\ 0.5072 & 0.6387 & 0.7903 & 0.3971 & 0.8663 & 0.3760 & 0.7133 \end{pmatrix} \quad (7)$$

Man beachte den Zusammenhang zwischen erster Spalte (Impulsantwort) und erster Zeile (2te bis 7te Eintragung in verkehrter Reihenfolge, nicht einfach FLIPUP transponiert!).

Eine typische zirkulante Matrix

Mittels farbiger Darstellung illustriert sehen Faltungsmatrizen wie folgt aus (wobei die Impulsantwort die erste Spalte darstellt):



Diagonalisierung von zirkulanten Matrizen

```
F7 = fft(eye(7)); D7 = F7' * CONVC * F7;
imagesc(abs(D7)); shg
```

illustriert die Tatsache, dass die Fourier-Matrix als Basis-Wechsel gedacht, eine (jede!) zirkulante Matrix diagonalisiert!

```
norm(D7 - diag(diag(D7))),
```

zeigt, dass $D7$ tatsächlich eine Diagonalmatrix ist!
`diag(diag(A))` ist der "reine Diagonalteil" der Matrix!

Weiters ist die Eintragung in der Diagonale genau die Fourier Transformierte des erzeugenden Vektors c:

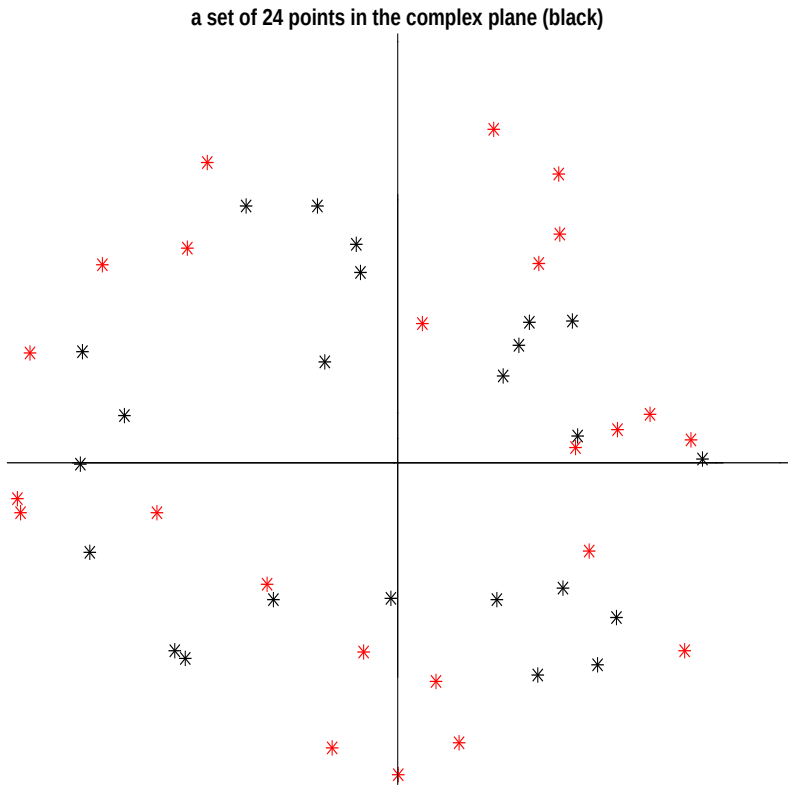
```
norm(fft(c(:))*7 - diag(D7))
```

Gibt es noch andere Fourier Transformationen?

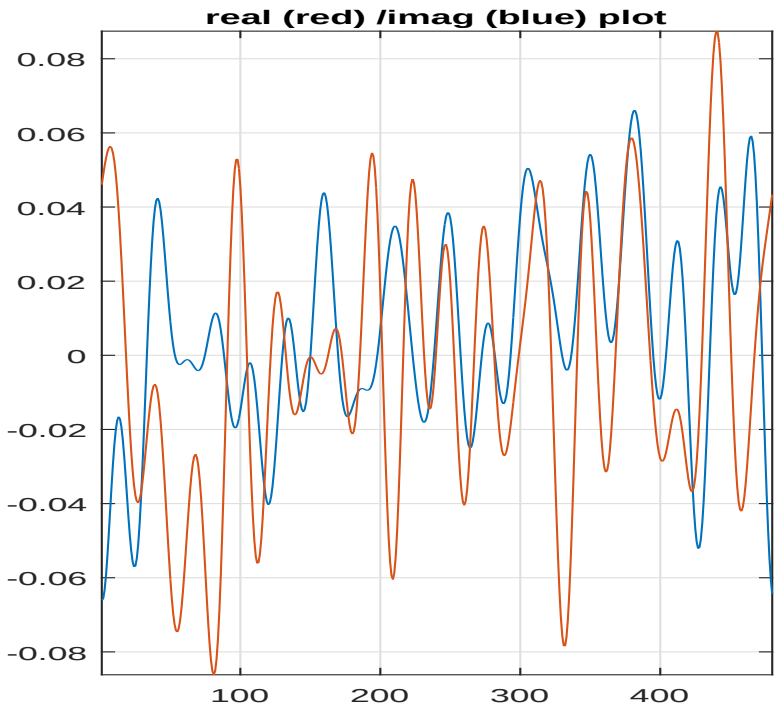
Die Frage lautet korrekt gestellt: Gibt es eventuell noch andere Orthormalbasen, welche eine Diagonalisierung von solchen Systemen erlaubt? Wenn ja, wie könnten sie aussehen? Wenn nein (richtige Antwort!), warum sehen sie so aus wie sie aussehen.
Im Detail ist diese Frage nicht mit einem MATLAB Experiment zu beantworten, aber immerhin plausibel zu machen, und zwar durch eine Illustration, was die Multiplikation mit komplexen Zahlen bedeutet, und daraus abgeleitet wie (und nur wie) eine beschränkte Funktion aussieht, die eine komplexe Zahl vom Absolutbetrag eins (aus dem Einheitskreis) als Eigenwert hat bzgl. des translationsoperators T_x: Es muss ein SPRIRALE sein, d.h. eine REINE FREQUENZ. Das kann man illustrieren!

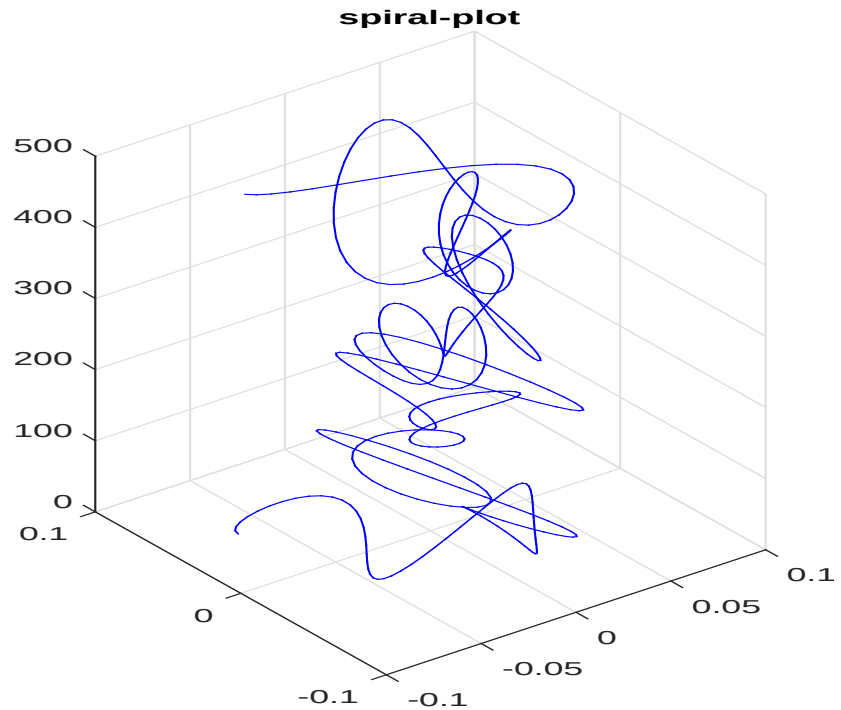
Multiplikation von komplexen Zahlen

Die rote Punktmenge ist durch eine komplexe Multiplikation aus der schwarzen hervorgegangen! Mit WELCHEM Faktor in $\mathbb{C}$? Erinnerung: Multiplikation entspricht einer Drehstreckung (Polardarstellung der komplexen Zahlen!).

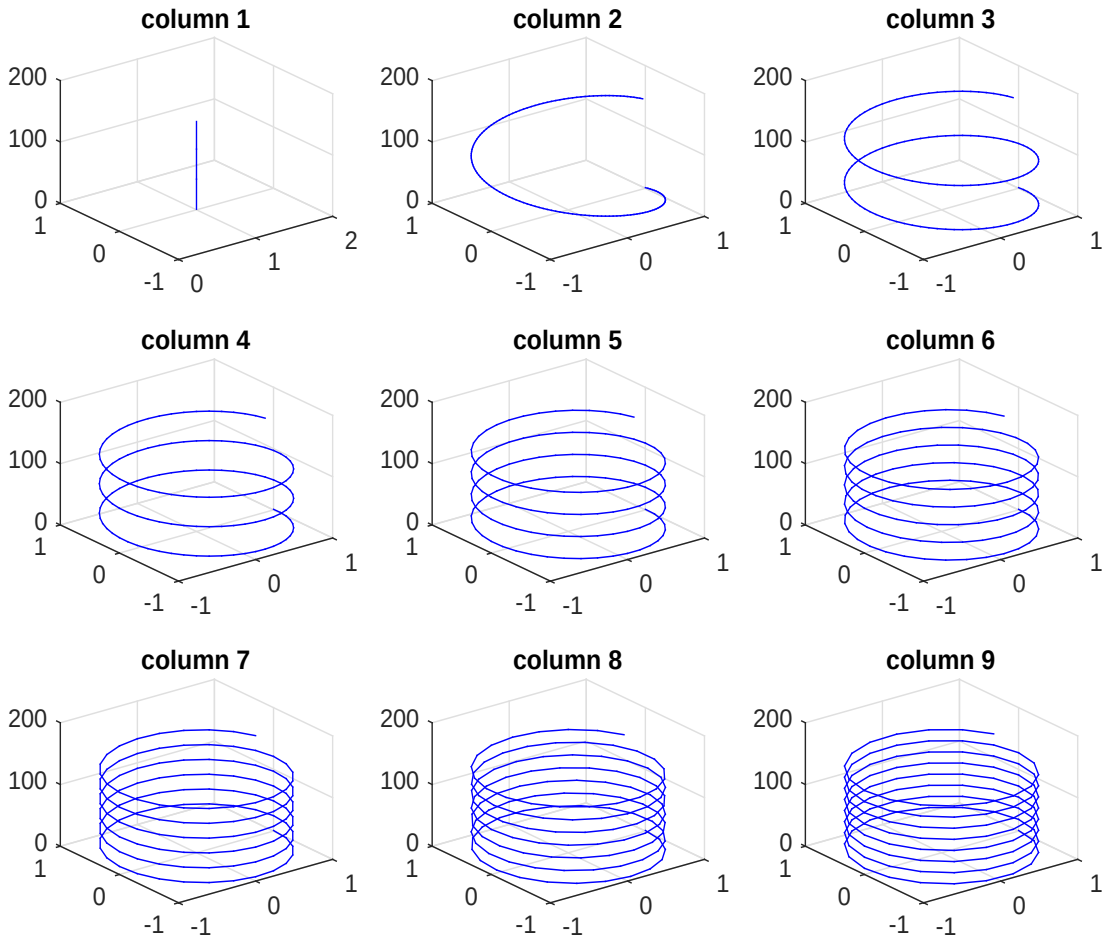


Plot von komplexwertigen Funktionen der Zeit





Spalten der Fourier Matrix: Spiralen



Fourier Basis ist kein Zufall

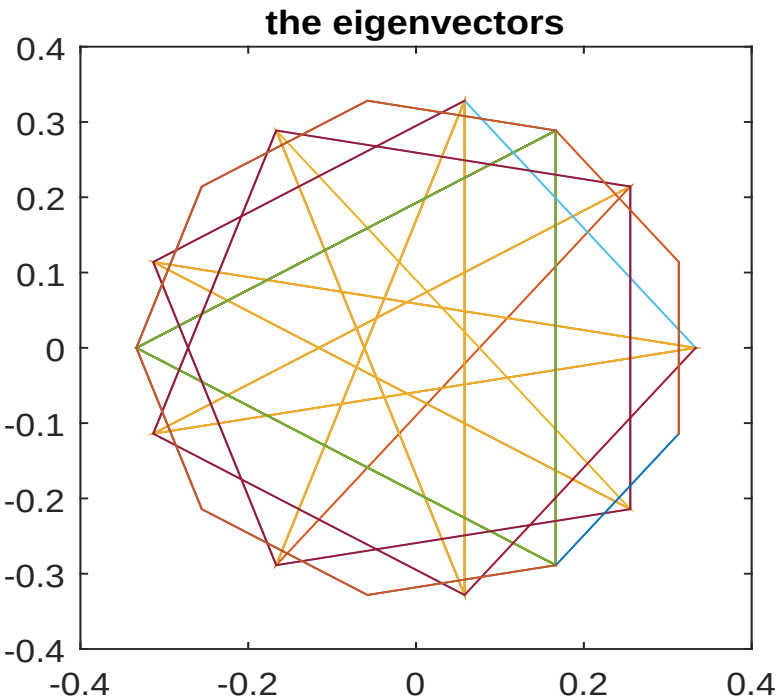
Auch ein Zufallsexperiment führt zu derselben Einsicht!

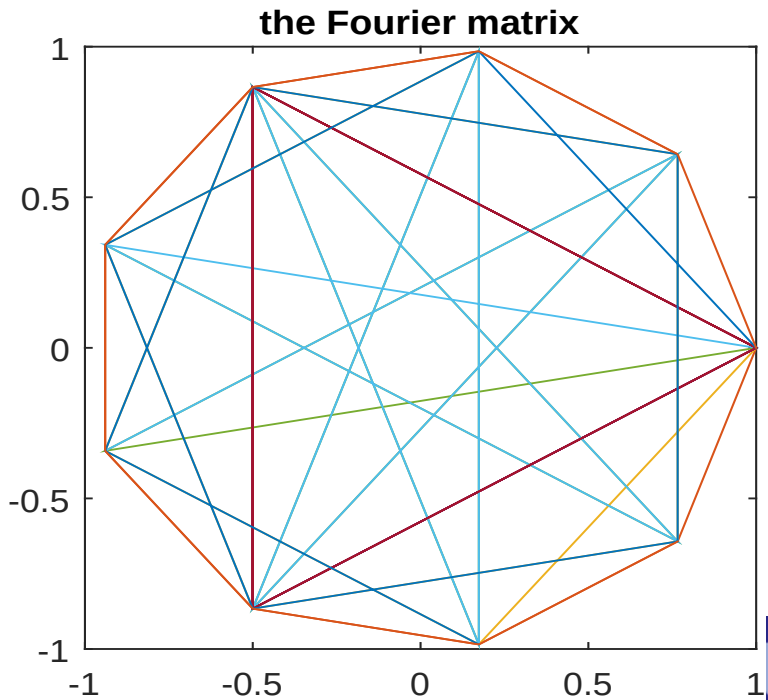
Es ist nicht nur so, dass die Fourier Basis aufgrund des Exponential-Gesetzes alle zyklischen Translationsoperatoren in Multiplikationsoperatoren überführt, sie diagonalisieren vielmehr *alle translationsinvarianten linearen Systeme*! Die Begründung: die System-Matrizen sind Summen von reinen Translationsmatrizen.

Es gilt aber auch die Umkehrung. Mehr oder minder jedes (zufällige) translationsinvariante linear System (m.a.W., jede positiv definite zirkulante Matrix) hat als Orthonormalbasis von Eigenvektoren (!!bis auf das Vorzeichen) genau die reinen Frequenzen!

Eigenvektoren von Zufalls-Systemen

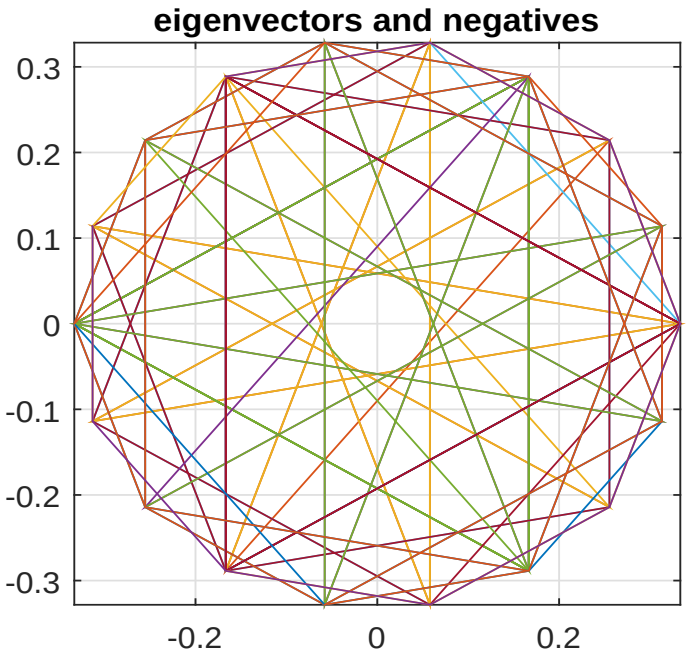
Visualisieren wir die Eigenvektoren einer solchen Matrix (der Einfachheit halber f.d. Fall $N = 9$):

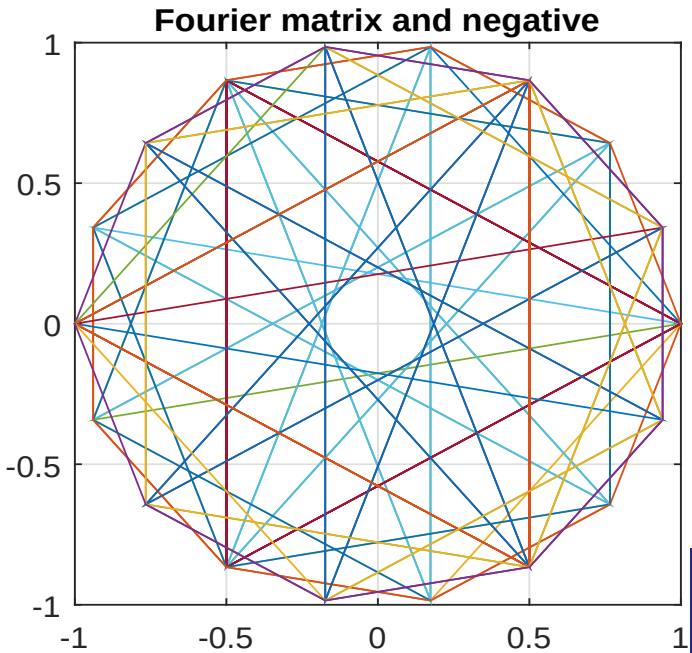




Eigenvektoren von Zufalls-Systemen

Da Eigenvektoren jeweils nur bis auf das Vorzeichen eindeutig bestimmt sind, plotten wir diese (in $\mathbb{C}$) zusammen mit ihren negativen Varianten (nun für $N = 17$)!





Praktische Fragen

Die verschiedenen Varianten zeigen **enorme Zeitunterschiede!**

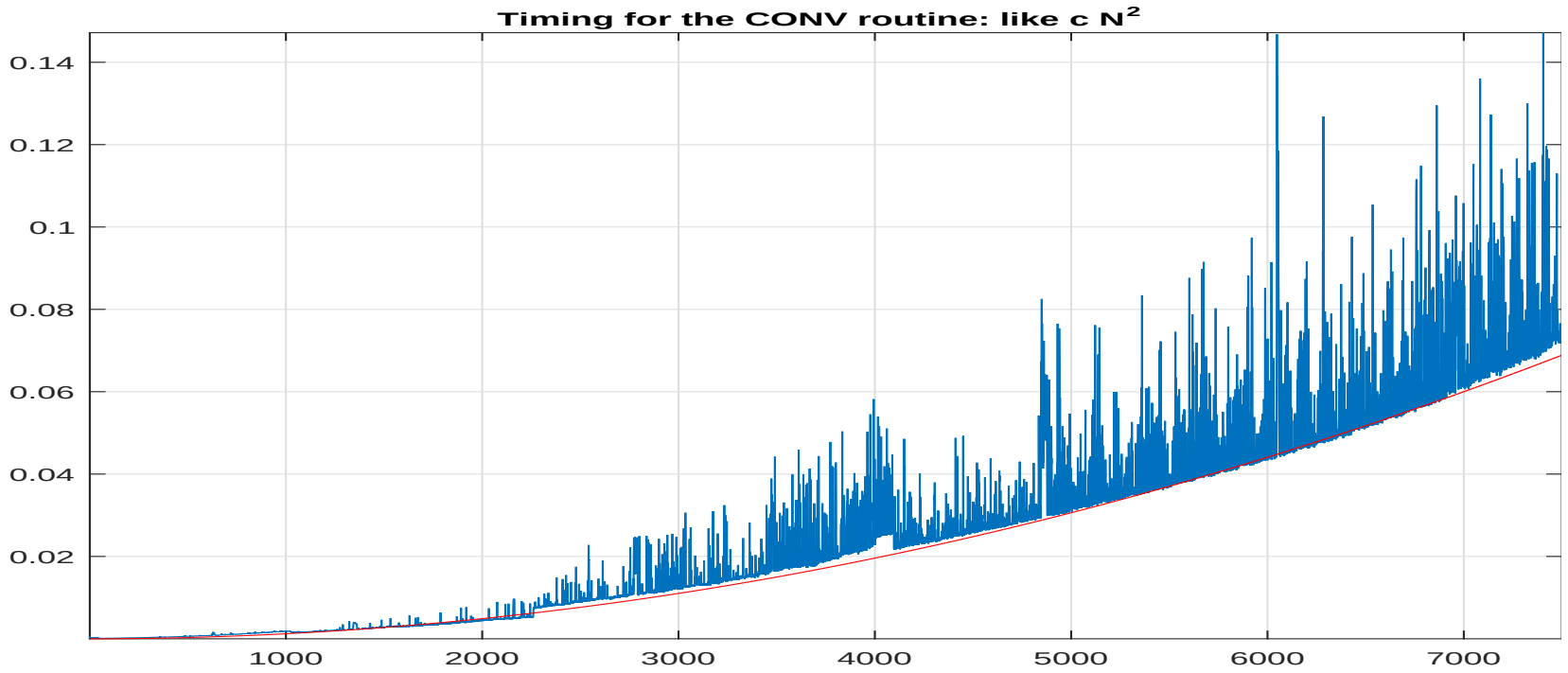
```
>> xx = rand(1,4410); d.h. 1/10 Sekunde Audiosignal
>> F = fft(eye(4410));
>> tic; myDFT(xx); toc; :NAIVE IMPLEMENTATION
>> tic; F*xx(:); toc :MATRIX MULTIPLIKATION
>> tic; fft(xx); toc :FFT ROUTINE
```

```
Elapsed time is 130.442276 seconds.
Elapsed time is 0.038570 seconds.
Elapsed time is 0.000194 seconds.
```

Zur Kontrolle:

```
>> yx = myDFT(xx); norm(yx - fft(xx))
ans = 1.8122e-09
```

Dauer des CONV-Befehls für wachsendes N



Wichtige Eigenschaften der FFT/DFT

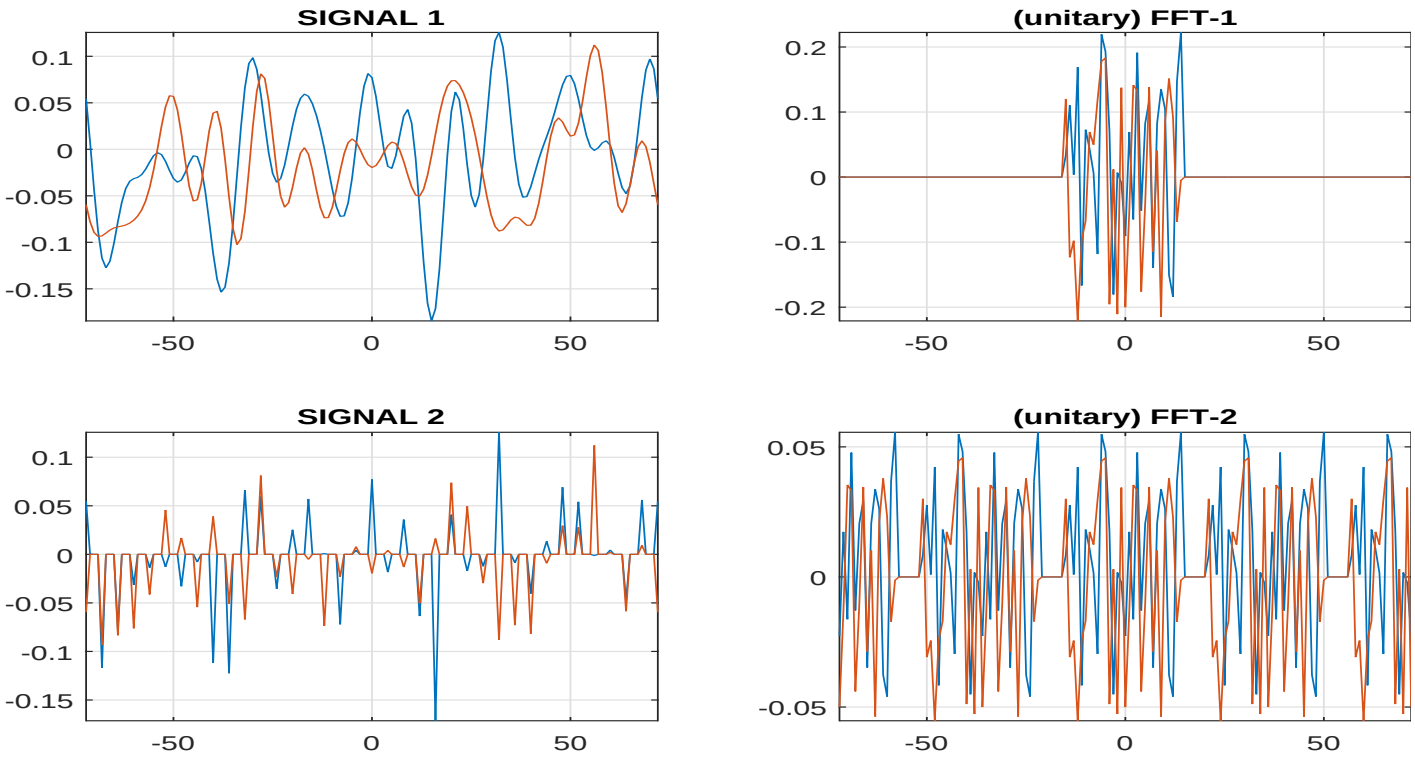
Da eine (zyklische) Verschiebung der Koeffizienten einfach eine Erhöhung der Potenz bedeutet, entspricht dies eine Multiplikation der Fourier Koeffizienten mit einer reinen Frequenz.

Ebenso ist die Prozedur des *downsampling* gut erklärbar. Ersetzt man die Folge $[c_1, \ldots, c_{2n}]$ durch die Folge

$$[c_1, 0, c_3, 0, c_5, \ldots c_{2n-1}, 0]$$

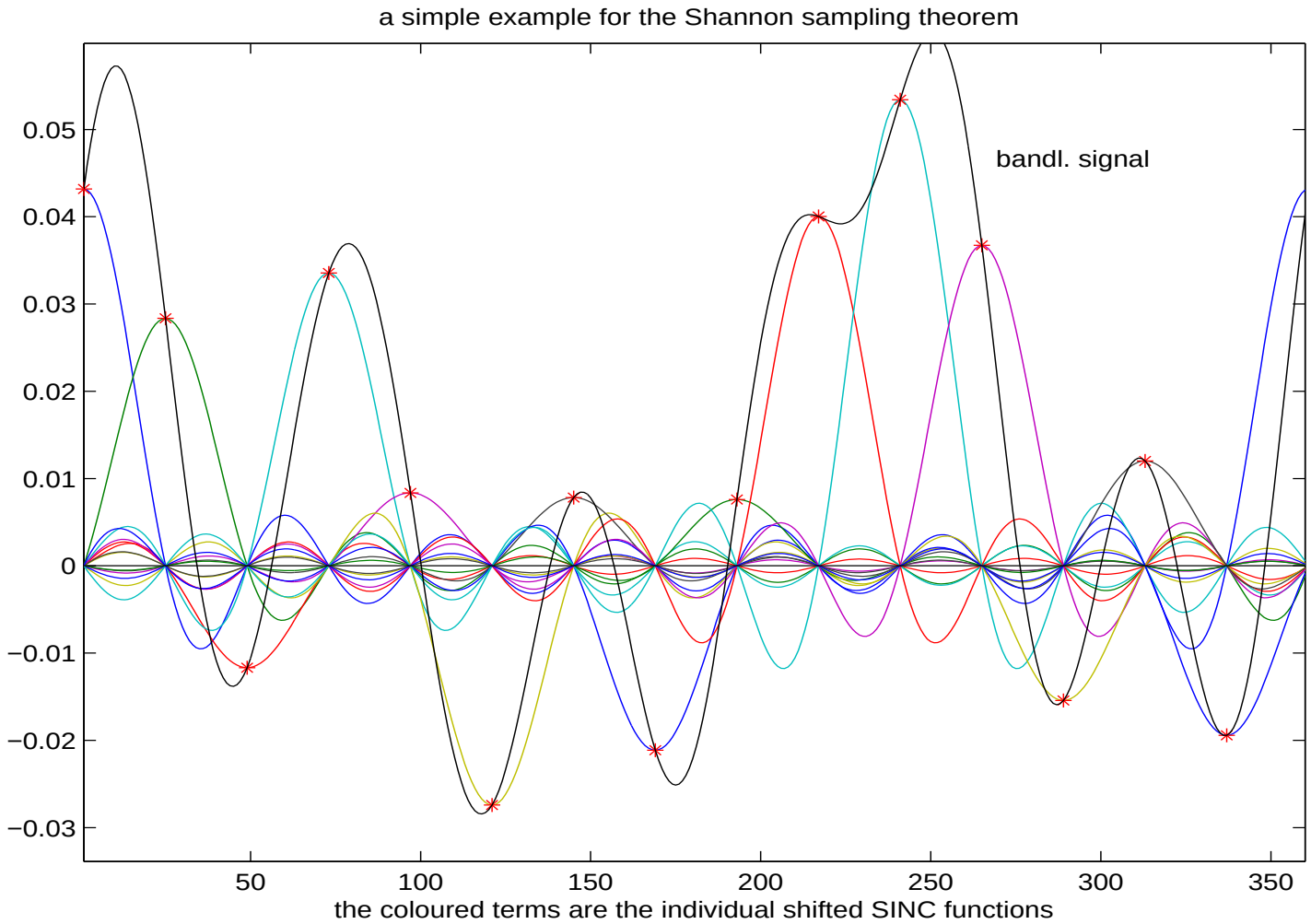
so entspricht die Anwendung der FFT Routine der Auswertung eines Polynoms $q(z)$. Setzt man $d_k := c_{2k-1}, 1 \leq k \leq n$ so gilt offenbar $p(z) = q(z^2)$, mit $q(y) = d_1 + d_2 y + \ldots d_n y^{n-1}$.

Shannon Abtast Theorem



Wie man *sieht* kann man den zentralen Block (rund um Null) durch Multiplikation mit einem Filter zurückgewinnen.

Rekonstruktion mittels Shifted SINC-Funktionen



Typische Fehler and Tricks

Es gibt einige Bücher und Hinweise in der angewandten Literatur, die auf "Probleme" und spezielle Tricks hinweisen, die notwendig sind, wenn man von der kontinuierlichen FT zur diskreten FT übergeht.

"Angeblich" herrschen im Diskreten "andere Verhältnisse"!??

Beispielsweise kann man sich fragen, inwieferne die wichtige Eigenschaft der normierten Gauss-Funktion $g_0(t) := e^{-\pi|t|^2}$, nämlich invariant unter der (Integral-Form der) Fourier Transformation zu sein, also $\mathcal{F}(g_0) = g_0$ zu erfüllen, ein Analogon in der endlich-diskreten Situation zu haben.

Richtiges Abtasten und Approximation durch Vektoren

Wir haben gesehen, dass die "naive Form" der Anwendung der FFT *nicht zu einer FFT-invarianten* diskreten Gauss-Folge führt (wir sehen einen Plot in der komplexen Ebene!).

Auch das Prinzip, dass "symmetrische Folgen" reelle FFTs haben sollten, muss mit Vorsicht behandelt werden. Was heißt es für eine Folge $(c_k)_{k=1}^N$ symmetrisch bzw. hermitsch zu sein?

Eine zirkulante Matrix ist genau dann hermitsch wenn:
$c_1 \geq 0$, und $c_{k+1} = conj(c(N+1-k))$ für $1 \leq k \leq N/2$.

REZEPT: **Sampeln und Periodisieren**, idealerweise Samplingrate $1/k$ und Periode k, mit $k \in \mathbb{N}$.

CONV und Cauchy-Produkt der Koeffizienten

Die Polynomfunktionen bilden nicht nur ein Vektorraum, welcher auch noch invariant bzgl. Translationen und Streckungen ist, sie bilden auch eine *kommutative Algebra*.

Die Bedeutung des *Grades eines Polynoms* liegt darin, dass sich beim Multiplizieren der Polynomfunktionen die Grade addieren!

Kubische Polynomfunktionen sind klarerweise vom Grad $= 3$, bzw. von der *Ordnung* (Zahl der Koeffizienten) $= 4$. In der

Beschreibung auf der Ebene der Koeffizienten (CONV!) kann die Multiplikation mit Hilfe des sogenannten **Cauchy-Produktes** charakterisiert werden (ausmultiplizieren und neu anordnen!)

$$c_k = \sum_{k=0}^{r+s} a_k b_{r+s-k}.$$

Der CONV-Befehl und das PASCAL Dreieck

```
>> format compact >> aa = [1,1]; bb = aa;
>> for jj=1:5; bb = conv(bb,aa), end
bb = 1 2 1
bb = 1 3 3 1
bb = 1 4 6 4 1
bb = 1 5 10 10 5 1
bb = 1 6 15 20 15 6 1
```

Liefert offenbar die Koeffizenten des Polynoms mit den Koeffizienten $[1,1]$, also $p(x) = 1 \cdot x + 1 = x + 1$ für polyval(aa, x), bzw. $(x+1)^k, k = 2, \ldots, 6$.

Unkonventionelle Anwendungen

1. Multiplikation von sehr langen Zahlen (Polynome an der Stelle $t = 10$);
2. Summen von zufälligen Variablen mit diskreten Verteilungen;
3. Demonstration des zentralen Grenzwertsatzes; Binomial-Verteilungen exakt;
4. Multiplikation und Division von Polynomen in 2 Variablen (mittels FFT2);

Unkonventionelle Anwendungen Ia

Die ja offenbar das Ergebnis des MATLAB-Befehls

$$\text{polyval}([1, 2, 3], 10)$$

gerade die natürliche Zahl 123 ist, ist klar, **dass man jeder natürlichen Zahl (Zifferfolge) eine Folge von Koeffizienten zuordnen kann, sodass das entsprechende Polynom an der Stelle $t = 10$ gerade diese natürliche Zahl ergibt.**

Die MULTIPLIKATION von POLYNOMEN entspricht dann genau der Multiplikation der entsprechenden ganzen Zahlen, ist aber mittels FFT schnell realisierbar! Lediglich am Ende muss noch bereinigt werden! 17 Zehner sind 7 Zehner und 1 Hunderter!

Unkonventionelle Anwendungen Ib

Eine studentische Arbeit hat die entsprechenden Routinen (Addition, Multiplikation, Potenzen, Division, etc.) realisiert:

```
>> bigmul('3434543534678 85','57987923434293487124 2587')
ans = 199162047520704871884305106362548818495
>> bigpow('34343',9)
>> ans = 66457510529646949647733745761756398845543
```

DEMO zur Routine "cleanvec":

```
>> bin = zeros(1,9); bin(1:2) = 1;
>> bin8 = round(real(ifft(fft(bin).^ 8)))
bin8 = 1 8 28 56 70 56 28 8 1
>> cleanvec(bin8)
ans = 2 1 4 3 5 8 8 8 1
```

Unkonventionelle Anwendungen II

Wir haben schon angedeutet, dass man die Wahrscheinlichkeitsverteilung für das Würfel-Experiment mit dem Polynom

$$q(x) = (x^6 + x^5 + x^4 + x^3 + x^2 + x)/6$$

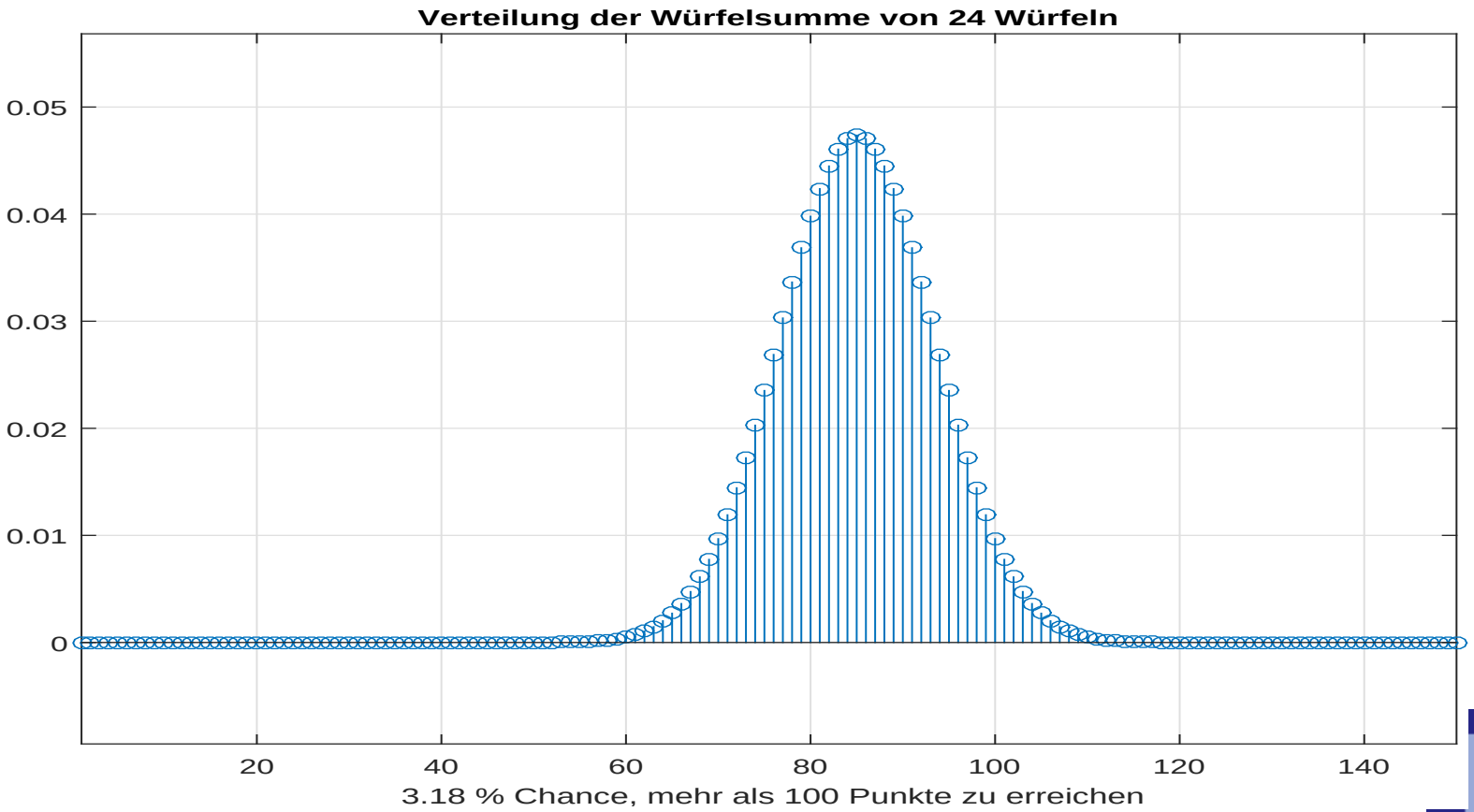
identifizieren kann.
Die Wahrscheinlichkeitsverteilung für die Summe von 2 Würfeln haben wir schon gesehen: $q(x)^2 = p(x)^2/36$ (s. oben).

Die entsprechende Verteilung für 20 Würfel:

```
w = zeros(1,150); w(2:7) = 1/6; w24 =
real(ifft(fft(w).^ 24)); sum(w24(101:150)), stem(w24)
```

Das Ergebnis zeigt, dass die Chance, mit 24 Würfeln mindestens 100 Punkte zu erzielen bei etwa $3,18\%$ liegt.

Unkonventionelle Anwendungen IIb



Unkonventionelle Anwendungen III

Das Quadrieren (Multiplizieren) von Polynomen in zwei Variablen kann man mit Hilfe der FFT2 bewerkstelligen:
Nehmen wir $p(x, y) = 2 + x - y - 2xy + x^2$; Die entsprechenden Koeffizienten kann man leicht in eine Matrix M übertragen. Da klar ist, dass die höchste Potenz von $p^2(x, y)$ natürlich x^4 ist, genügt es, die Daten in eine 5×5-Matrix einzubetten.

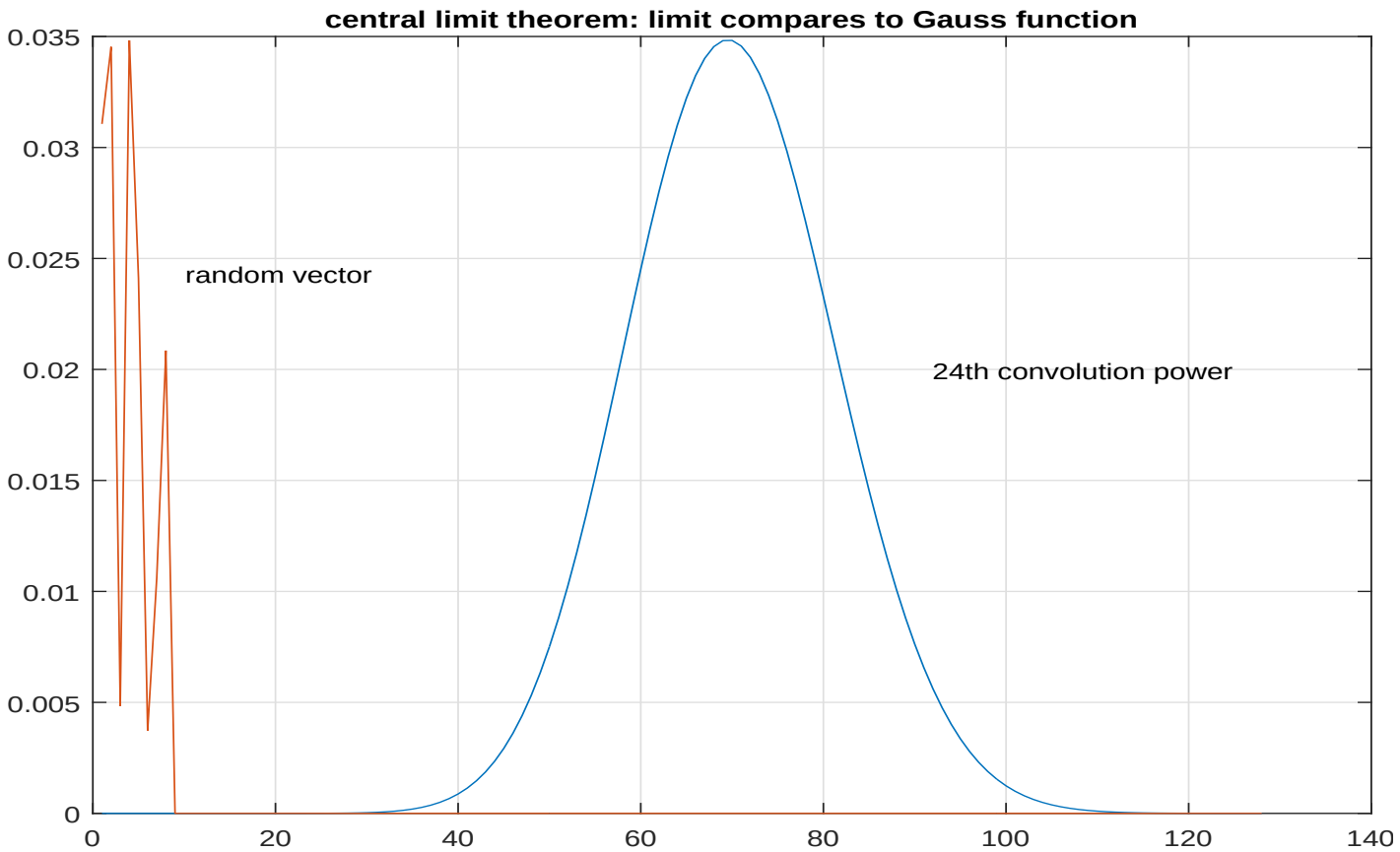
$$M = zeros(5); M(1,1) = 2; M(1,2) = 1; M(2,1) = -1;$$

$$M(2,2) = -2; M(1,3) = 1; round(real(ifft2(fft2(M).^2)))$$

Daraus kann man die Koeffizienten von $p^2(x, y)$ ablesen.
Der Faktor von x^2y ist -6 ist, oder der von x^3 ist 2.
Auch Division von Polynomen kann so bewerkstelligt werden!

Unkonventionelle Anwendungen IV

Der zentrale Grenzwertsatz besagt, dass eine (hohe) Faltungpotenz immer einer Gauss-Funktion ähnlich sieht:



Nicht-periodische Funktionen: die STFT

Normalerweise wird argumentiert, dass man bei der Behandlung von *nicht-periodischen* Signalen integrierbare oder wenigstens quadratisch integrierbare Funktionen nehmen muss. Die übliche Argumentation betrachtet Fourier-Reihen-Entwicklungen mit immer dichterem Spektraum (die Grund-Periode wird größer!). Das Ergebnis ist eine durch in Integral definiert koninuierliche Fourier-Transformation (definiert für $f \in \boldsymbol{L}^1(\mathbb{R}^d)$):

$$\hat{f}(s) = \int_{\mathbb{R}^d} f(t)\, e^{2\pi i\, s \bullet t}\, dt, \quad s \in \mathbb{R}^d.$$

In diesem Rahmen kann (Satz von Plancherel) die FT als unitärer Automorphismus von $\left(\boldsymbol{L}^2(\mathbb{R}^d), \|\cdot\|_2\right)$ aufgefaßt werden, mit

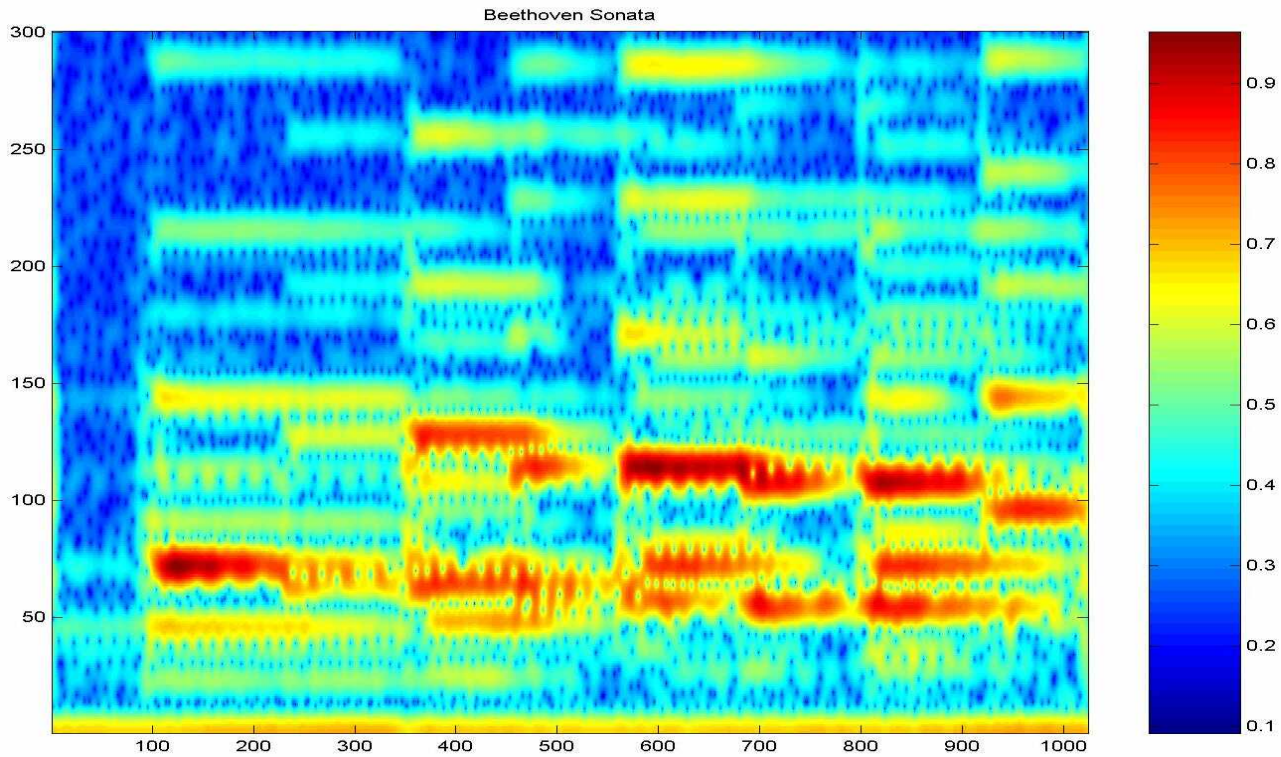
$$\|f\|_2 = \|\hat{f}\|_2, \quad f \in \boldsymbol{L}^2(\mathbb{R}^d).$$

Kurzzeit-Fourier-Transformation und Anwendungen

Aber diese Verallgemeinerung der endlichen Fourier-Transformation enthält keinerlei Information, wann welches Sound-Ereignis stattfindet. Kurz gesagt, es erlaubt keine Aussage, über die *Melodie* die dem analysierten Signal zugrundelag. Dies kann nur mit Hilfe der sogenannten **Kurzzeit-Fourier-Transformation** bzw. **Sliding Window Fourier Transform** bewerkstelligt werden. Auch diese läßt sich mit Hilfe von MATLAB gut illustrieren.

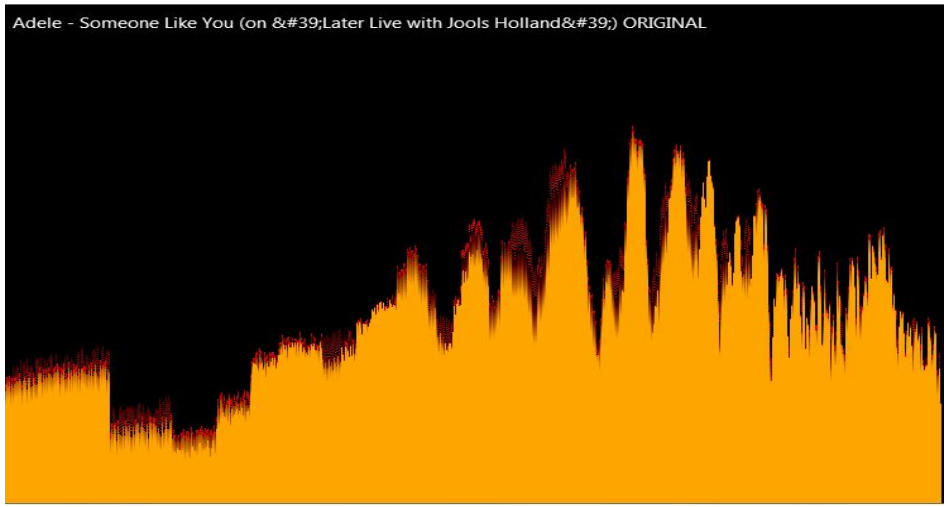
Es gibt zwei Interpretationen der sog. **Gabor Analysis**. Entweder geht es um eine Rekonstruktion einer STFT aus einer diskret abgetasteten Version, andererseits geht es um eine *atomare Darstellung* von Signalen mit Hilfe von zeit-frequenz-verschobenen Varianten eines *Gabor Atoms*.

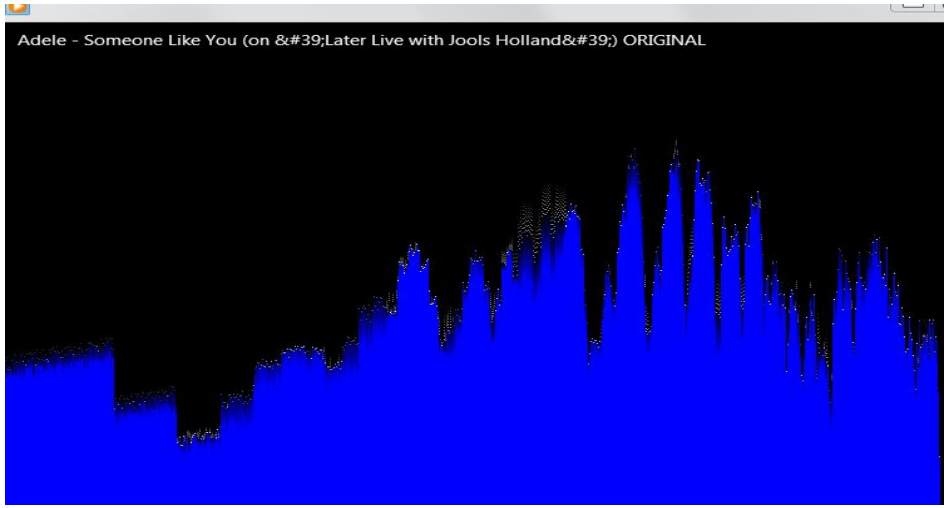
Gabor Analysis: Beethoven Piano Sonata



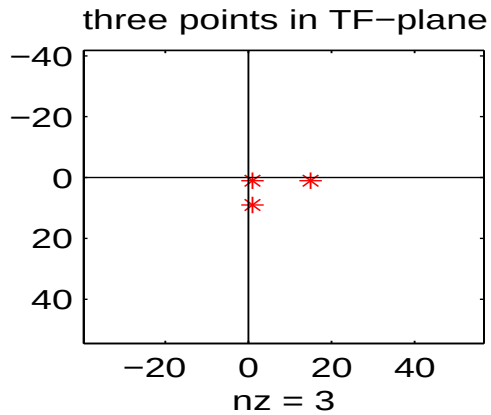
Das Spektrogramm einer Beethoven Sonate!

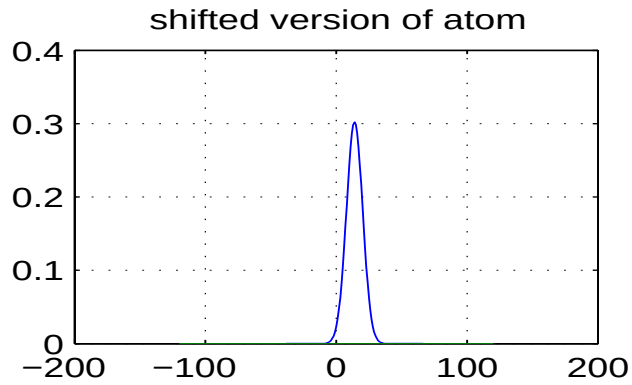
Visualisierungen mittels Windows Media Player

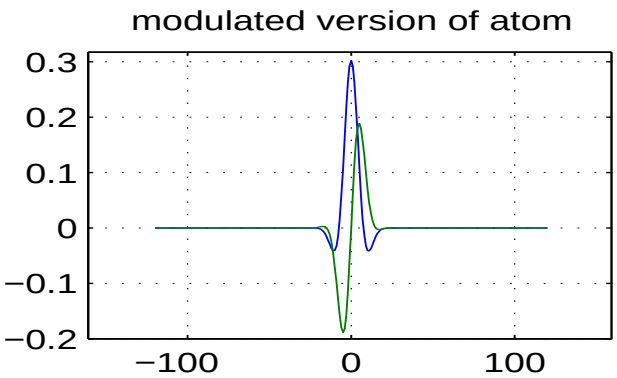


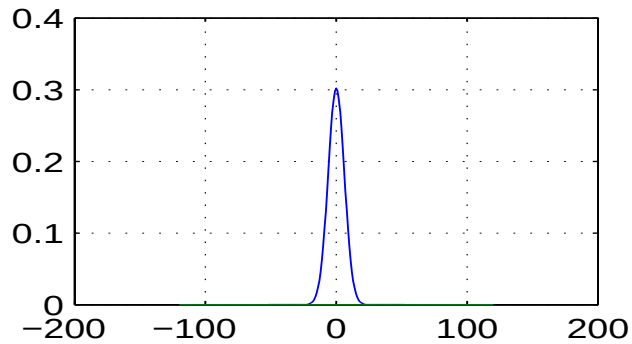


Zeit-Frequenz Verschiebungen

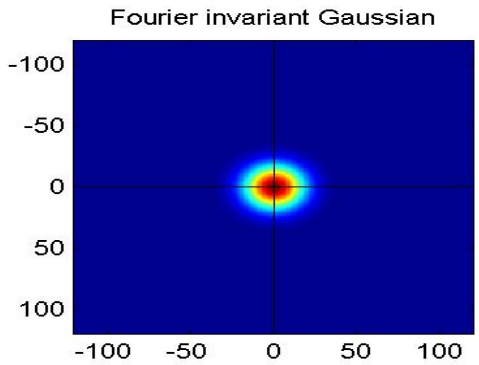


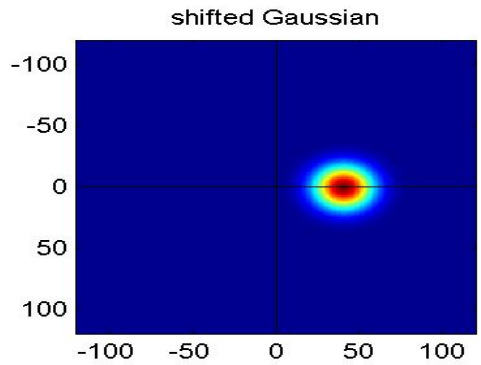


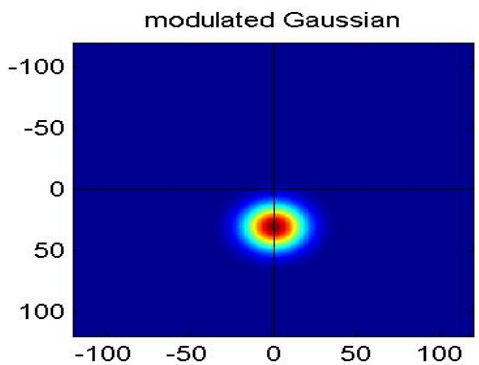


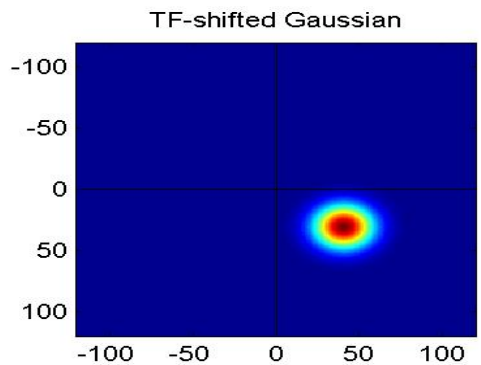


Zeit-Frequenz Verschiebungen im Spektrogramm

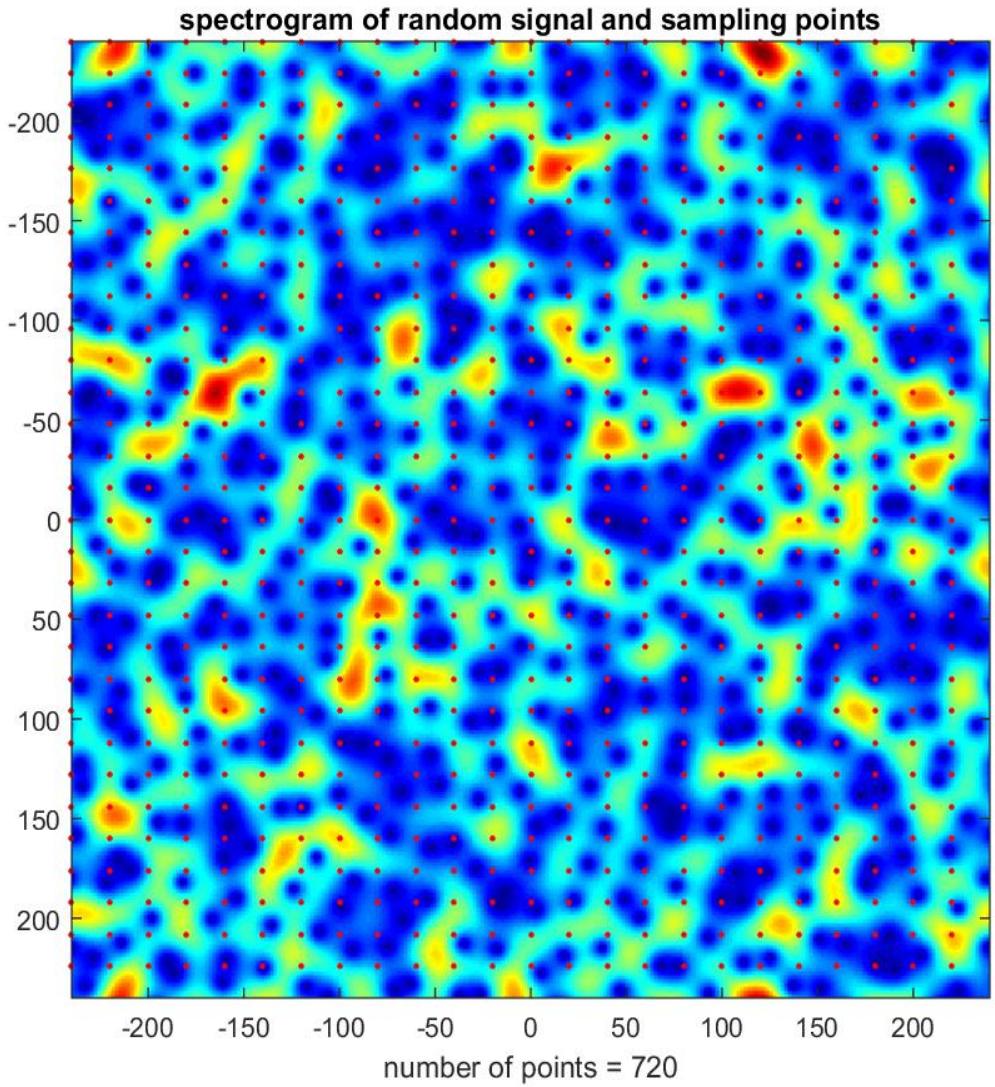






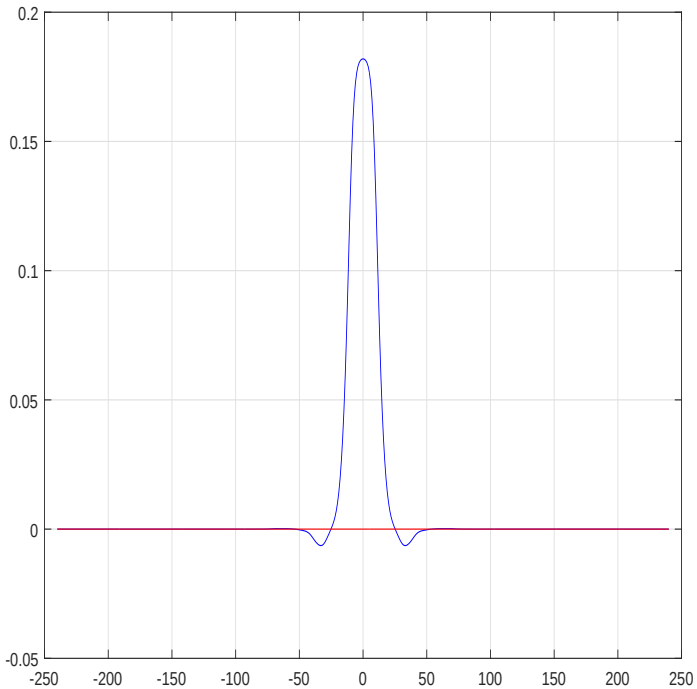


Spektrogramm und Sampling-Punkte



Spektrogramm und Gabor Reihen-Darstellung

Mathematisch können wir die Situation so beschreiben: Wählt man das Atom g passend, etwa so:



Tight Gabor Frames, Fortsetzung

Dann ist die *Gabor Familie* $(\pi(\lambda)g)_{\lambda\in\Lambda}$ ein "tight Gabor frame", d.h. mit der Notation $g_\lambda := \pi(\lambda)g$ hat man die Darstellung

$$f = \sum_{\lambda\in\Lambda} \langle f, g_\lambda\rangle g_\lambda, \quad f \in \mathbb{C}^n.$$

Bis auf einen festen Faktor $C = C(\Lambda, g)$ gilt: die Länge des Koeffizienten-Vektors $(\langle f, g_\lambda\rangle)$ entspricht der Norm $\|f\|_2$.

Solche "tight frames" sind natürlich nicht gleich einem Orthonormalsystem, weil sie linear abhängig (redundant!) sind (sie sind jedoch für viele Zwecke gleichwertig!).

Sog. *Gabor Multiplier* operieren (multiplikativ) auf den Gabor Koeffizienten, und sind die Basis des MP3-Verfahrens (kombiniert mit dem psychoakustischen Maskierungseffekt).

Resourcen im Internet und Links

Wir werden eine Zusammenstellung der Routinen, die im Laufe des Vortrages sowie die verwendeten NuHAG M-files zur Verfügung stellen. Entsprechende Anfragen bitte an den Autor richten: `hans.feichtinger@univie.ac.at`

Sie sollen auch unter `www.nuhag.eu` im Internet bereitgestellt werden, und zwar unter der Adresse

www.nuhag.eu/EXPO10

MATLAB Material der Gruppe NuHAG findet man allgemein unter
http://www.univie.ac.at/nuhag-php/mmodule/

LTFAT, die **The Large Time-Frequency Analysis Toolbox** findet man unter der Adresse
http://ltfat.sourceforge.net/

Acknowledgement

Dieser Vortrag wurde im Frühjahr 2016, während eines Gastaufenthaltes des Autors am Institut für Mathematik der Universität Salzburg verfasst. Während dieser Zeit erhielt der Autor partielle Unterstützung durch das FWF Projekt (Fond zur Förderung der Wissenschaftlichen Forschung) **F5504-N26**, welches Teil des Forschungsschwerpunktes "Quasi-Monte Carlo Methods: Theory and Applications" ist. Der Autor dankt seinem Gastgeber, Prof. Peter Hellekalek.
Einige der Ideen, die in diesem Vortrag enthalten sind, entstanden durch Diskussionen mit meinem Dissertanten Bayram Ülgen, dessen Dissertation mit dem Titel *"Signal- und Bildverarbeitung im Kontext Höherer Schulen. Grundlegende Vorstellungen und Chancen aus der Sicht der Mathematikdidaktik"* in wenigen Wochen verteidigt werden wird.

Distributionen-Theorie und Banach Gelfand Triple

Die in diesem Vortrag vorgestellten Eigenschaften der Fourier Transformation bzw. der Zeit-Frequenz Analyse kann relativ einfach in den Kontext von kontinuierlichen Signal übertragen, wenn man sich auf geeignete Räume von Testfunktionen beschränkt.
Der klassische Zugang verwendet den *nuklearen Frechetraum* $\boldsymbol{\mathcal{S}}(\mathbb{R}^d)$, der *rasch fallenden Funktionen*. Das ist ein spezieller Typ von topologischen Vektorräumen, mit einer abzählbaren Familie von Seminormen. Es gibt aber auch eine Variante, wo man mit einer einzigen Norm auskommt, nämlich die (spezielle) Segal Algebra mit folgender Norm:

$$\|f\|_{\boldsymbol{S}_0} := \|STFT_{g_0} f\|_{\boldsymbol{L}^1(\mathbb{R}^{2d})} < \infty,$$

für den man etwa die Fourier Inversionsformel einfach mit Riemannschen Integralen realisieren kann.

Das Banach Gelfand Triple

Zusammen mit $\boldsymbol{L}^2(\mathbb{R}^d)$ und dem Dualraum $\boldsymbol{S}_0'(\mathbb{R}^d)$ bilden die drei Räume ein sog. *Banach Gelfand Tripel* $(\boldsymbol{S}_0, \boldsymbol{L}^2, \boldsymbol{S}_0')(\mathbb{R}^d)$, welches eine für Ingenieur-Anwendungen adäquates mathematisch korrektes Hilfsmittel darstellt. Aber das ist eine andere Geschichte...
Man kann beispielsweise die Fourier Transformation als eindeutigen *unitären Banach Gelfand Triple Automorphismus* charakterisieren, welcher (so wie in der Linearen Algebra Situation) die "reinen Frequenzen" auf *Dirac Masze* abbildet.