# Modul 5033 Digitale Signalverarbeitung Dozent: Rolf Vetter

# 

### *Praktische Übung No 2*

# Schätzung von Signalkennwerten im Zeitbereich

### Ziel

* Einführung in Schätzung von Kennwerten und Kenngrössen im Zeitbereich, d.h. Mittelwert, Standardabweichung, Varianz, Verteilungsdichte und Korrelation
  + Festigung der Kenntnisse und Verknüpfung mit praktischer Anwendung
  + Kenntnisnahme der Grenzen einer Schätzung mit einer finiten Anzahl Samples
  + Umsetzung mittels Programmierung

### Aufgaben

### Programmierung der Varianz

1. Schreiben Sie eine Funktion (Aufruf [var\_x]=var\_for(x)), welche die Varianz eines Signals mittels einer for Schleife ermittelt.
   1. Schreiben Sie eine Testfunktion [var\_x,var\_x\_mat]=test\_var\_for(N), welche ein gaussverteiltes weisses Rauschen mit N Samples und einer Standardabweichung 10 erzeugt und seine Varianz mit var\_for() und der MATLAB Funktion var() berechnet. Überprüfen Sie, dass die Resultate identisch sind.

*Vorgesehene Dauer: 0.5 Lektionen*

1. Fakultative Übung: Schreiben Sie eine Funktion (Aufruf [var\_x]=var\_sp(x)), welche die Varianz eines Signals mittels Skalarprodukt ermittelt.
   1. Schreiben Sie eine Testfunktion [var\_x,var\_x\_mat]=test\_var\_sp(N), welche ein gaussverteiltes weisses Rauschen mit N Samples und einer Standardabweichung 10 erzeugt und die Varianz mit var\_sp() und der MATLAB Funktion var() berechnet. Überprüfen Sie, dass die Resultate identisch sind.

*Vorgesehene Dauer: 0.5 Lektionen*

### Streuung einer Schätzung

1. Das Ziel dieser Übung ist das Wahrnehmen, dass jede Schätzungsmethode in einem Umfeld mit Rauschen (stochastisch), einen Schätzungsfehler und somit eine Streuung (Varianz) hat. Dies kann mit einer sogenannten Monte-Carlo Simulation hervorgehoben werden. Wir werden es in dieser Übung anhand der Mittelwertschätzung illustrieren. Man schreibe daher ein Skript montecarlo\_mean, das die folgenden Befehle ausführt:

Nbr\_Sim=100; % Anzahl Schätzungen

N=16; % Anzahl Samples

a=0.1; % Gewichtung des Rauschens

for i=1:Nbr\_Sim

x=1+a\*randn(N,1);

mu\_x(i)=mean(x);

end

% Ausgabe der Resultate

disp(['Streuung (Varianz) der Schätzung=' num2str(var(mu\_x))])

disp(['Standardabweichung der Schätzung=' num2str(std(mu\_x))])

% Anzeige der Resultate

plot(mu\_x)

xlabel('Simulation N^o')

ylabel('Geschätzter Wert')

hold all

* 1. Führen Sie das Skript aus. Was stellen Sie fest?
  2. Geben Sie für a den Wert 1 ein und führen Sie das Skript erneut aus. Was stellen Sie fest? Analyse und Diskussion?
  3. Für einen Wert von a=1, verändern Sie die Anzahl Samples (z.B. 10,100,1000). Feststellungen, Analyse und Diskussion?

*Vorgesehene Dauer: 1 Lektionen*

### Verteilungsdichte

1. Man erzeuge ein gaussverteiltes weisses Rauschen x=randn(N,1) mit N Samples und stelle sein Histogramm mit der MATLAB Funktion hist(x,K) dar. Testen Sie die folgenden Parameterpaare

N=100000, K=100;

N=1000,K=100;

Feststellungen und Schlussfolgerungen?

*Vorgesehene Dauer: 0.5 Lektionen*

1. Fakultative Übung: Man programmiere und teste eine Funktion verteilungsdichte(x,K), welche die Verteilungsdichte des Signals x für K Stützpunkte ermittelt und darstellt. Man teste die Funktion mit einem gaussverteilten weissen Rauschen und vergleiche das Resultat mit der MATLAB Funktion hist(x,K). Analyse und Diskussion?

*Vorgesehene Dauer: 0.5 Lektionen*

### Autokorrelation

1. Man ermittle die Autokorrelation mit Hilfe der MATLAB Funktion xcorr() für folgende Signale:
   1. Sinussignal
   2. Kosinus-signal
   3. Zufallsbitsequenz
   4. Gaussverteiltes weisses Rauschen

Stimmen die ermittelten Schätzungen mit den theoretischen Werten überein?

Hinweis: Ermittlung und Anzeige von Autokorrelation siehe Signale und Systeme mit MATLAB Seite 3.9.

*Vorgesehene Dauer: 1 Lektionen*

1. Fakultative Übung: Programmieren Sie die Autokorrelation mittels Skalarprodukt. Testen Sie Ihre Funktion durch einen Vergleich mit der MATLAB Funktion xcorr.

Ergebnisse

# Aufgabe 1 und 2

Matlab Code für for-Loop, Skalarprodukt und Test:

|  |
| --- |
| %% Startbedingungen  N = 1000;  x = 10 \* randn(N, 1);    %% Variante For-Schleife  var\_x = 0;  for i = 1 : N  var\_x = var\_x + x(i)^2;  end  var\_x = var\_x / N;  disp(['Varianz mit For-Loop: ' num2str(var\_x)]);  disp(['Varianz mit ''var'': ' num2str(var(x))]);    %% Variante Skalarprodukt  var\_x = x'\*x/N;    disp(['Varianz mit Skalarprodukt: ' num2str(var\_x)]);  disp(['Varianz mit ''var'': ' num2str(var(x))]); |

Resultat:

**Varianz mit For-Loop: 101.4412**

**Varianz mit 'var': 101.4032**

**Varianz mit Skalarprodukt: 101.4412**

**Varianz mit 'var': 101.4032**

Die Abweichung zur Matlab Funktion „var“ beträgt bei beiden Methoden 0.04%. Da beide eigenen Implementationen im Prinzip genau die gleiche Mathematische Operation berechnen, ergeben sie auch das gleiche Resultat bzw. die gleiche Abweichung zur Matlab Funktion.

# C:\Users\Marcel\Pictures\Screenpresso\2013-03-08_10h34_32.pngAufgabe 3

* 1. Die Abweichung der Schätzung liegt bis zu 0.6 daneben.
  2. Blau: a = 0.1,

Streuung (Varianz) der Schätzung = 0.00080465

Standardabweichung der Schätzung = 0.028366

Grün: a = 1

Streuung (Varianz) der Schätzung=0.069388

Standardabweichung der Schätzung=0.26342

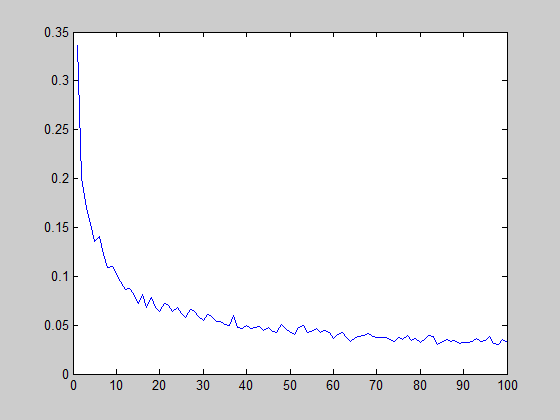
Man erkennt gut, dass die Schätzung mit dem Rauschen, das gleich gewichtet ist wie der gesuchte Wert, viel ungenauer (ca. Faktor 10)

Ist. Dies macht auch Sinn, weil wir das additive Rauschen ja um Faktor 10 vergrössert haben.

Wenn das Verhältnis vom Signal zu a (dem Faktor des additiven Rauschens) gegen eins geht, kann man den Mittelwert mit einer einzigen Schätzung nicht mehr genau bestimmen.

* 1. Ändert man die Anzahl Samples, nimmt die Standardabweichung der Schätzung ab. Die Schätzung der Varianz nimmt mit 1/N ab, daraus folgt, dass die Standardabweichung mit 1/sqrt(N) abnimmt.

Die Methode ist somit praktisch um abzuschätzen, wie viele Samples genügen, um den Fehler genügend gut einzudämmen.



# Aufgabe 4

Matlab Code:

|  |
| --- |
| %% Histogramm mit N = 100'000 und K = 100  N = 100000;  K = 100;  x=randn(N,1);  figure  hist(x,K);    %% Histogramm mit N = 1'000 und K = 100  N = 1000;  K = 100;  x=randn(N,1);  figure  hist(x,K); |

Resultat:

|  |  |
| --- | --- |
| D:\BFH\DSV\Übung 2\histogramm_N_1000.bmp  K = 100, N = 1‘000 | D:\BFH\DSV\Übung 2\histogramm_N_100000.bmp  K = 100, N = 100‘000 |

Man sieht, dass die Gauss-Verteilkurve viel schöner sichtbar ist, wenn die Anzahl Samples viel grösser ist als die Anzahl bins ist. Das heisst bei wenigen Samples wird die Rauschfunktion noch nicht schön normalverteilt sein und deshalb werden Messungen, die auf statistischen Kennwerte der Rauschfunktion basieren, nicht genau sein.

# Aufgabe 5

Fakultativ – wegen mangelnder Zeit ausgelassen.

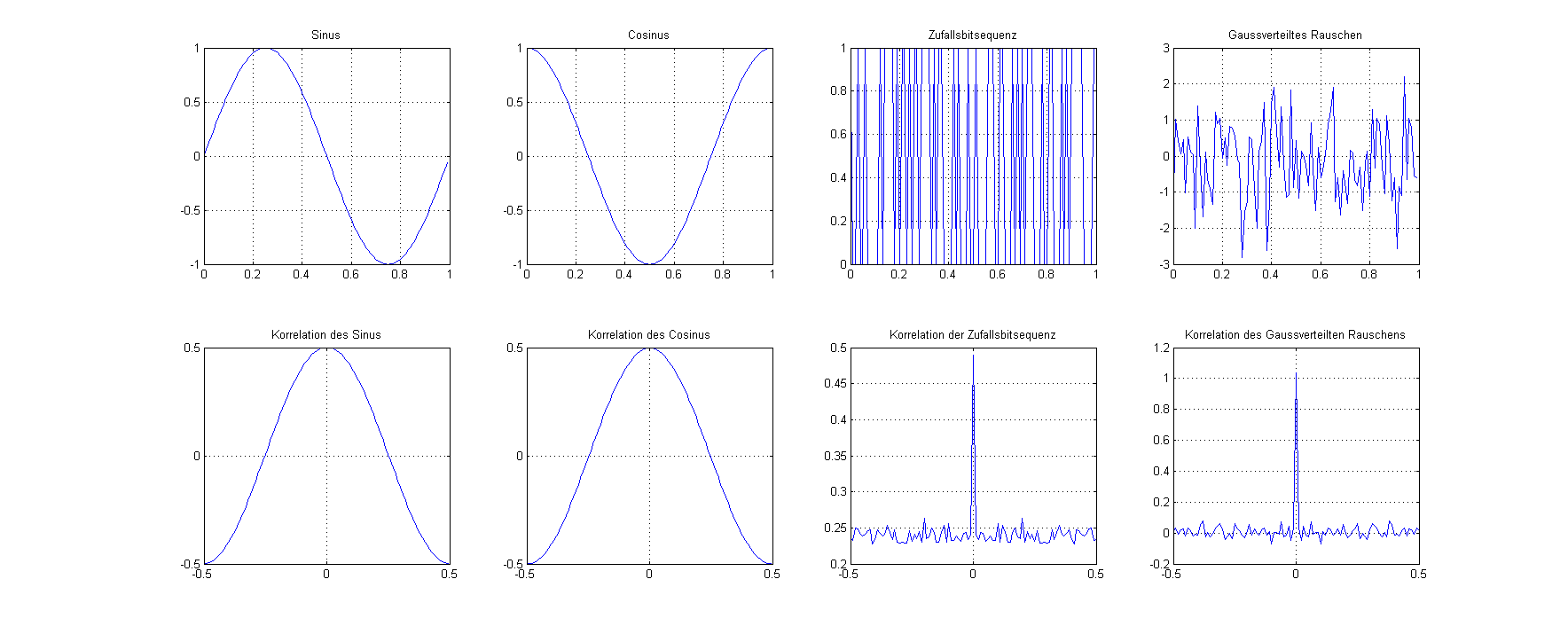
# Aufgabe 6

Wir haben die Signale Sinus, Kosinus, eine Zufallsbitsequenz und ein gaussverteiltes Rauschen einer Autokorrelation unterzogen.

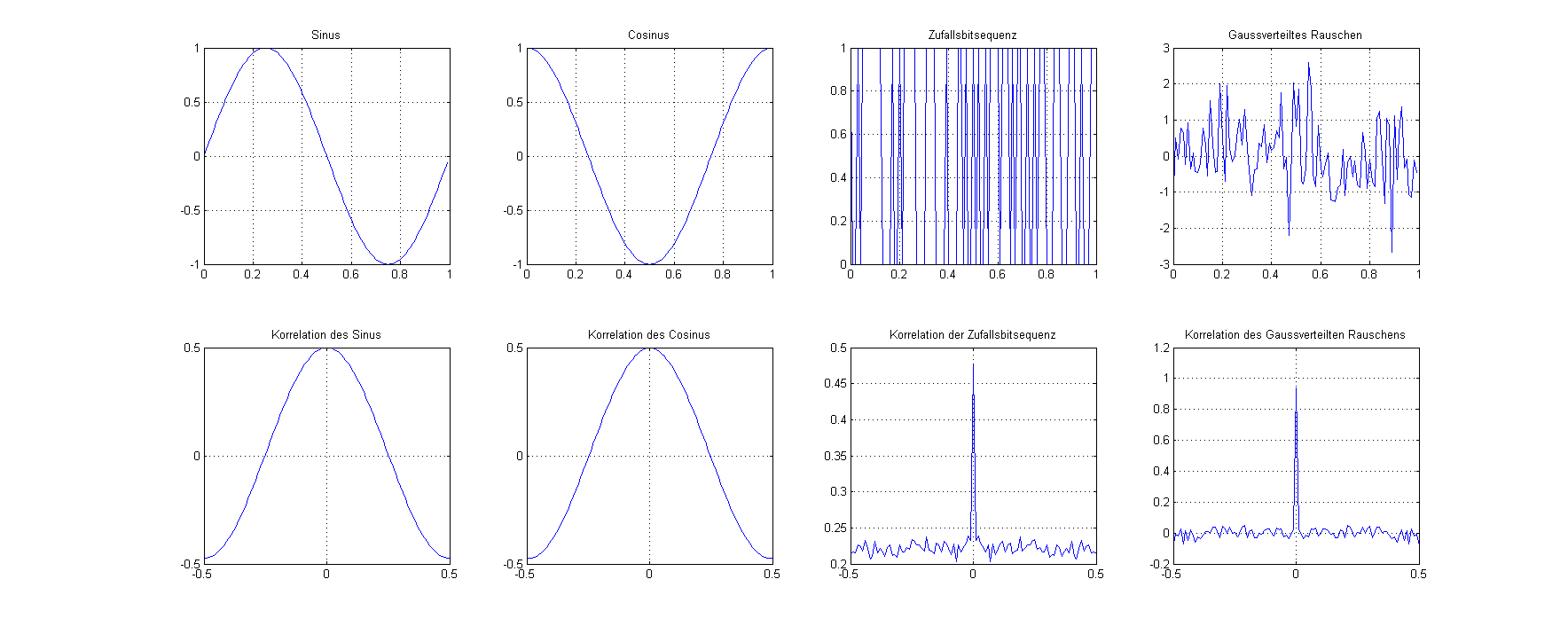
Matlab Code für eine Periode (biased und unbiased):

|  |
| --- |
| x = 0:0.01:(10 - 0.01);  x\_xcorr = -(length(x(:,1:50))-0)\*0.01:0.01:(length(x(:,1:50))-0)\*0.01;  f = 1;  opt = 'biased'; % oder 'unbiased'  sinus = sin(2\*pi\*f\*x);  cosinus = cos(2\*pi\*f\*x);  bitsequence = randi(2,1,length(x)) - 1;  noise = randn(1, length(x));      subplot(2,4,1); plot(x(:,1:100), sinus(:,1:100)); title('Sinus'); grid on;  subplot(2,4,5); plot(x\_xcorr, xcorr(sinus, 50, opt)); title('Korrelation des Sinus'); grid on;    subplot(2,4,2); plot(x(:,1:100), cosinus(:,1:100)); title('Cosinus'); grid on;  subplot(2,4,6); plot(x\_xcorr, xcorr(cosinus, 50, opt)); title('Korrelation des Cosinus'); grid on;    subplot(2,4,3); plot(x(:,1:100), bitsequence(:,1:100)); title('Zufallsbitsequenz'); grid on;  subplot(2,4,7); plot(x\_xcorr, xcorr(bitsequence, 50, opt)); title('Korrelation der Zufallsbitsequenz'); grid on;    subplot(2,4,4); plot(x(:,1:100), noise(:,1:100)); title('Gaussverteiltes Rauschen'); grid on;  subplot(2,4,8); plot(x\_xcorr, xcorr(noise, 50, opt)); title('Korrelation des Gaussverteilten Rauschens'); grid on; |

**Resultat: *unbiased*, eine Periode**



**Resultat: *biased*, eine Periode**

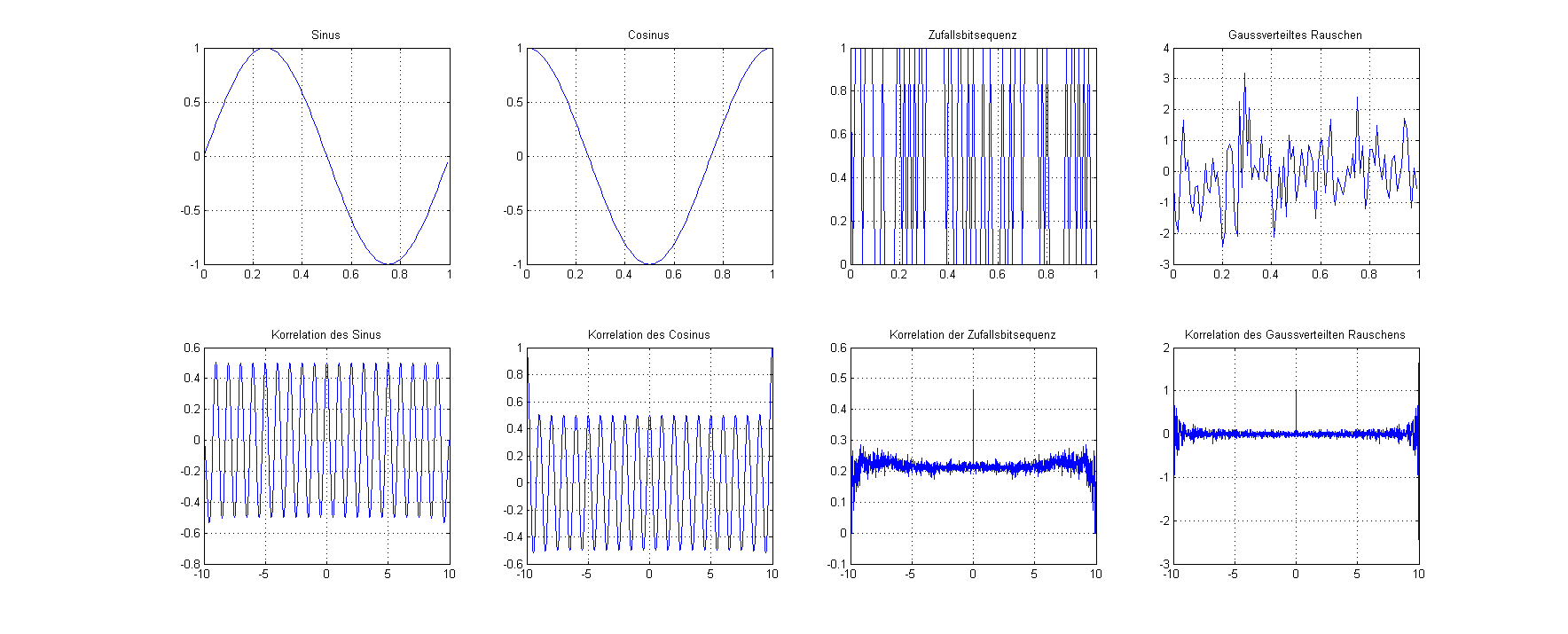


Man sieht genau die Signalformen, die theoretisch erwartet wurden. Der Unterschied zwüschen ‚*biased*‘ und ‚*unbiased*‘ ist aber fast nicht erkennbar. Deshalb müssen wir bei der Autokorrelation den gesamten Samplingbereich darstellen.

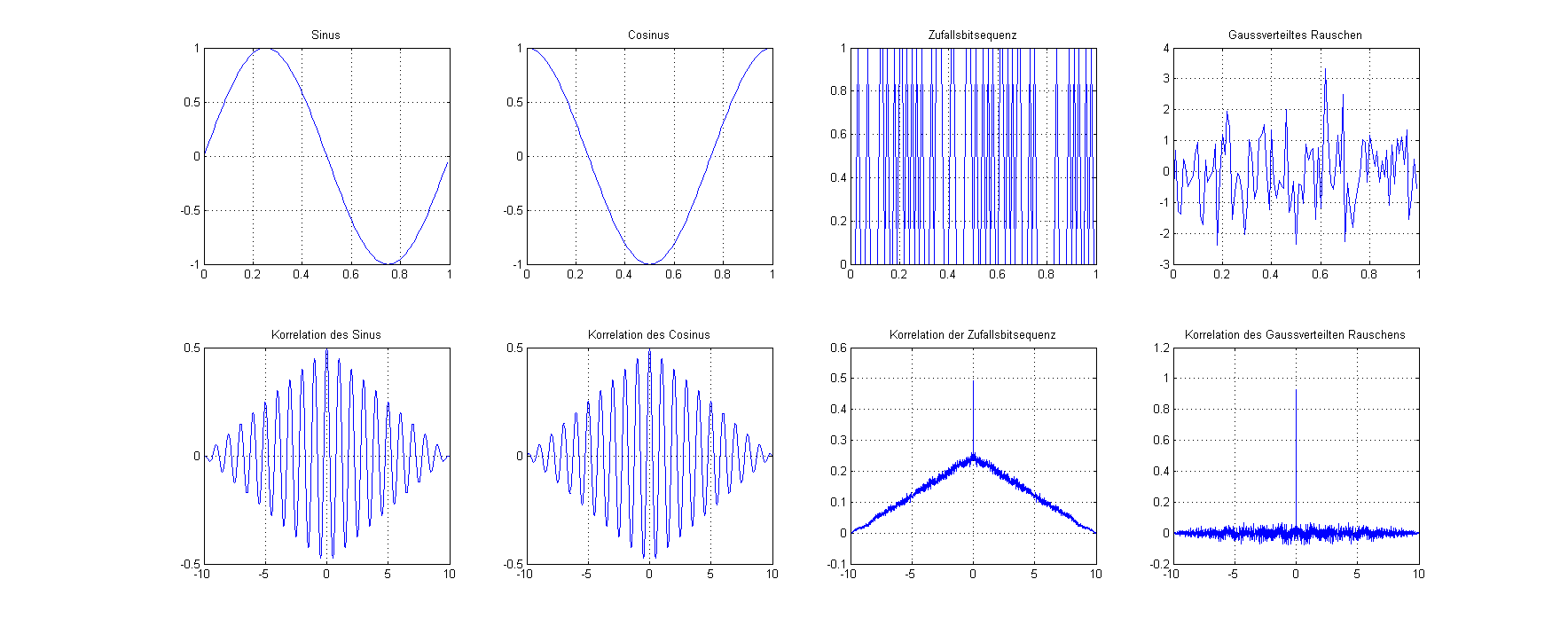
Matlab Code:

|  |
| --- |
| x = 0:0.01:(10 - 0.01);  x\_xcorr = -(length(x)-1)\*0.01:0.01:(length(x)-1)\*0.01;  f = 1;  opt = 'biased'; % oder 'unbiased'    sinus = sin(2\*pi\*f\*x);  cosinus = cos(2\*pi\*f\*x);  bitsequence = randi(2,1,length(x)) - 1;  noise = randn(1, length(x));      subplot(2,4,1); plot(x(:,1:100), sinus(:,1:100)); title('Sinus'); grid on;  subplot(2,4,5); plot(x\_xcorr, xcorr(sinus, opt)); title('Korrelation des Sinus'); grid on;    subplot(2,4,2); plot(x(:,1:100), cosinus(:,1:100)); title('Cosinus'); grid on;  subplot(2,4,6); plot(x\_xcorr, xcorr(cosinus, opt)); title('Korrelation des Cosinus'); grid on;    subplot(2,4,3); plot(x(:,1:100), bitsequence(:,1:100)); title('Zufallsbitsequenz'); grid on;  subplot(2,4,7); plot(x\_xcorr, xcorr(bitsequence, opt)); title('Korrelation der Zufallsbitsequenz'); grid on;    subplot(2,4,4); plot(x(:,1:100), noise(:,1:100)); title('Gaussverteiltes Rauschen'); grid on;  subplot(2,4,8); plot(x\_xcorr, xcorr(noise, opt)); title('Korrelation des Gaussverteilten Rauschens'); grid on; |

**Resultat: *unbiased*, 10 Perioden:**



**Resultat: *biased*, 10 Perioden:**



Hier sieht man nun die Wirkung des Biasing. Die Form des Überlagerten Dreiecks ist gut erkennbar.

Skalarprodukt Version der Autokorrelation:

|  |
| --- |
| function x\_corr=xcorr\_scalar(x)  x\_corr = zeros(1,length(x));  for k = 0:(length(x)/2-1)  x\_corr(k+1) = x(:,(k+1:end))\*x(:,(1:end-k))';  end    x\_corr = [fliplr(x\_corr(1:end/2)) x\_corr(2:end/2+1)];  end |

LABELS IMMER AUCH MIT ANGEBEN (X UND Y)