Stochastische Prozesse

May 24, 2016

1 Stochastische Prozesse

1.1 Stochastischer Prozess

Ein stochastischer Prozess ist eine Folge von Zufallsvariablen x_t.

1.2 White-Noise Prozess

Ein White-Noise-Prozess oder reiner Zufallsprozess ist eine identisch verteilte Zufallsvariable ϵ_t, die unabhängig von vorherigen Zeitschritten $t - \Delta t$ ist.

Beispiel Normalverteilung $\epsilon_t \sim N(\mu, \sigma)$

1.3 Autoregressive Prozesse

Beim autoregressiven Prozess (AR-Prozess) hängt die Zufallsvariable von vorherigen Zeitschritten ab. Im Fall der ersten Ordnung (AR1-Prozess) hängt der Zeitschritt t nur vom vorherigen Zeitschritt $t-1$ ab:

$$x_t = \alpha x_{t-1} + \epsilon_t$$

Der Koeffizient α wird Feedback-Parameter genannt.

1.4 Random-Walk Prozess

Der Spezialfall des AR1-Prozesses mit $\alpha = 1$ wird Random Walk oder auch Zufallsbewegung genannt.

$$x_t = x_{t-1} + \epsilon_t$$

1.5 Korrelation und Autokorrelation

Der Korrelationskoeffizient r zwischen N Beobachtungspaaren zweier Variabler x und y ist gegeben als

$$r(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Die Autokorrelation $r(x,l)$ ist die Korrelation der Zeitserie x_t mit der um die Distanz (Lag) l verschobenen gleichen Zeitreihe x_{t+l}. Besondere Bedeutung hat der Autokorrelationskoeffizient für $l = 1$.

1.6 Filter

1.6.1 Lineare Filter

Ein linearer Filter L transformiert eine Zeitreihe x_t in einer andere Zeitreihe y_t mittels der Berechnung

$$y_t = \sum_i a_i x_{t-i}$$

$$y_t = Lx_t$$

Die Folge der Koeffizienten (a_i) des linearen Filteroperators L wird auch als Impulsantwortfunktion bezeichnet.

1.6.2 Gleitender Mittelwert

Der gleitende Mittelwert oder auch Moving Average (MA) berechnet sich aus

$$y_t = \frac{1}{3}(x_{t-1} + x_t + x_{t+1})$$

Die sogenannte Fensterlänge (Window Size) beträgt in diesem Fall $W = 3$ (Anzahl der Koeffizienten).

1.6.3 Kausale Filter

Wenn das Filter nur von seinen gegenwärtigen und vergangenen Eingangswerten abhängt, wird es kausales Filter genannt.

Das Mittelwertfilter $y_t = \frac{1}{3}(x_{t-1} + x_t + x_{t+1})$ ist nicht-kausal, da der gegenwärtige Ausgangswert y_t vom zukünftigen Eingangswert x_{t+1} abhängt.

1.7 Stationarität

Die Funktionen Mittelwert $\mu(t) = E[x_t]$, Varianz $\sigma^2(t) = Var[x_t]$ und Kovarianz $\gamma(s,t) = Cov[x_s, x_t]$ seien jeweils über einen endlichen Fensterbereich W zu berechnen.

Die stochastische Zeitreihe x_t wird mittelwert- bzw. varianzstationär genannt, wenn der Mittelwert $\mu(t)$ bzw. die Varianz $\sigma^2(t)$ zeitlich konstant ist. Die Zeitreihe heißt kovarianzstationär, wenn $\gamma(s,t)$ nur von $s - t$ abhängt. Ist eine Zeitreihe mittelwert-, varianz und kovarianzstationär, so wird sie schwach stationär genannt.

2 AR1-Spektrum

Das korrespondierende Spektrum $G(\omega)$ (spektrale Dichte) eines Markov-Prozesses erster Ordnung ist gegeben als (Jenkins und Watts, 1968)

$$G(\omega) = \frac{F}{1 + \alpha^2 - 2\alpha\cos(\omega\Delta)}$$

mit der Frequenz ω und der (konstanten) spektralen Energiedichte F des Rauschantriebs.

3 Beispiele

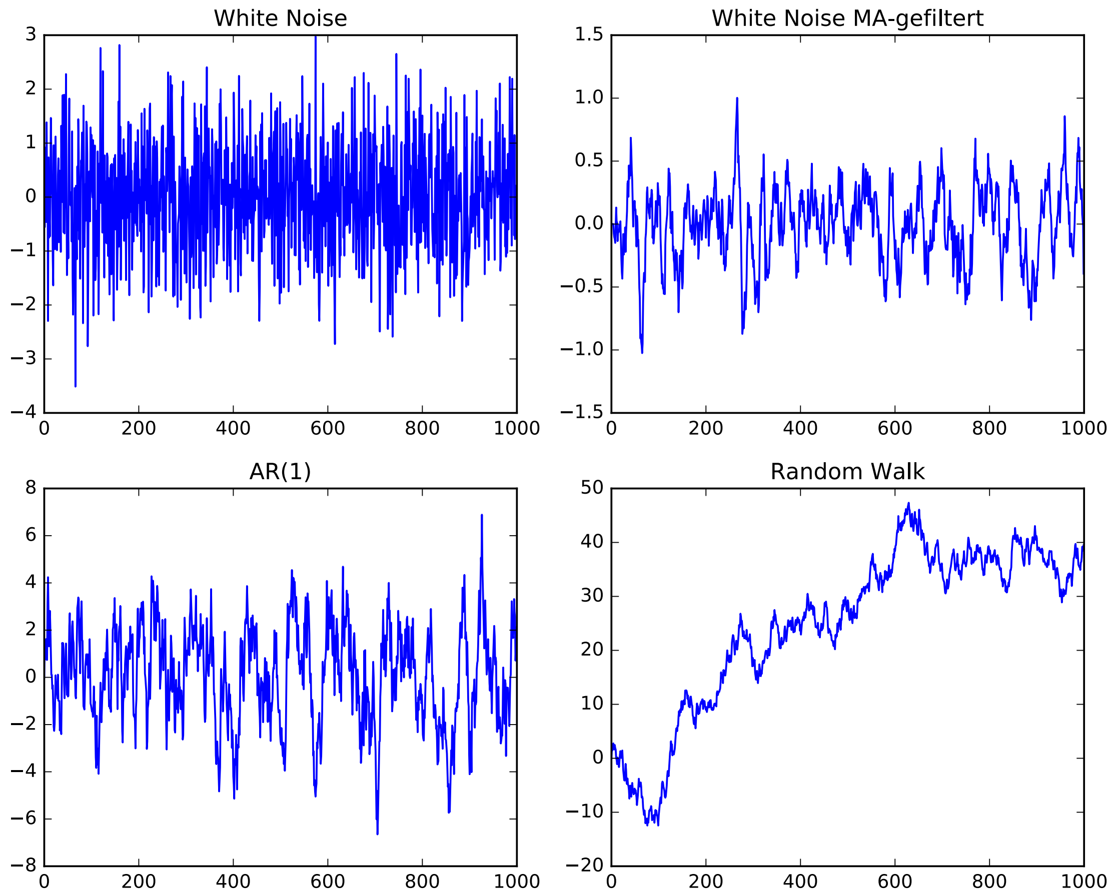
3.1 Stochastische Prozesse

In [24]:
```
%pylab inline
%config InlineBackend.figure_format = 'svg'

def AR1(N,a,b):
    Y=zeros(N)
    for i in range(1,N):
        Y[i]=Y[i-1]*a+b*randn(1)
    return Y

def MA_filter(Y,v):
    Y_in=Y
    Y=Y.copy()
```

```
    for i in range(1,N-1):
        Y[i]=1.0/(2*v+1)*(sum(Y_in[i-v:i+v+1]))
    return Y


N=1000
Y_white_noise=AR1(N,0,1)
Y_L=MA_filter(Y_white_noise,5)
Y_AR1=AR1(N,0.9,1)
Y_random_walk=AR1(N,1.0,1)

#####################################
figure(figsize=(10,8))
subplot(2,2,1)
plot(Y_white_noise)
title('White Noise')
subplot(2,2,2)
plot(Y_L)
title('White Noise MA-gefiltert')
subplot(2,2,3)
plot(Y_AR1)
title('AR(1)')
subplot(2,2,4)
plot(Y_random_walk)
title('Random Walk')
```

```
Populating the interactive namespace from numpy and matplotlib
```

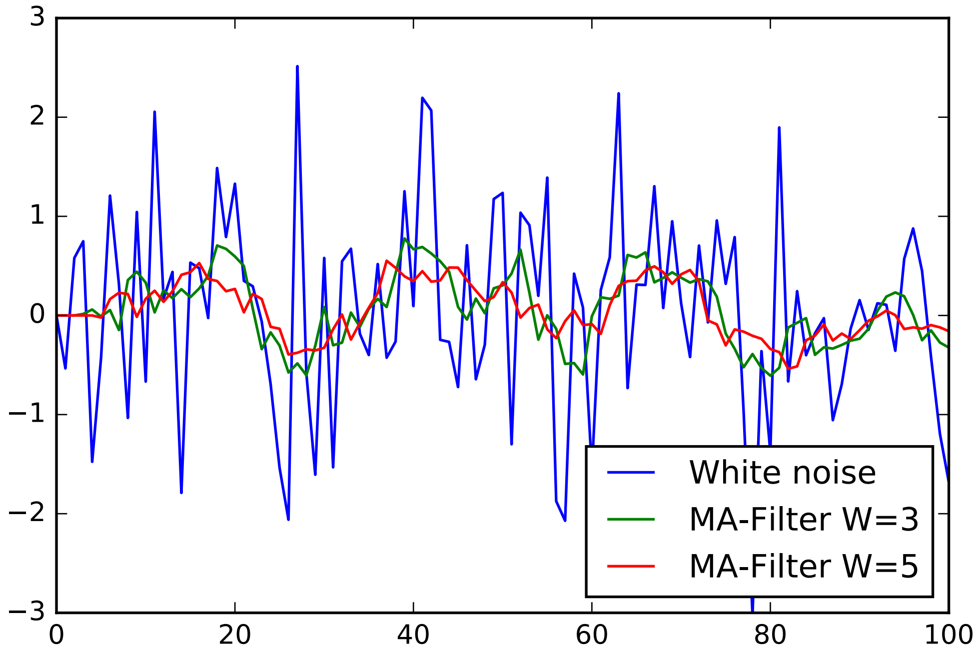
```
Out[24]: <matplotlib.text.Text at 0x7f47a7246b38>
```



3.2 MA-Filter

In [25]:
```
N=1000
Y_white_noise=AR1(N,0,1)
Y_L_W3=MA_filter(Y_white_noise,3)
Y_L_W5=MA_filter(Y_white_noise,5)

plot(Y_white_noise,label='White noise')
plot(Y_L_W3,label='MA-Filter W=3')
plot(Y_L_W5,label='MA-Filter W=5')
legend(loc=4)
axis([0,100,-3,3])
```

Out[25]:
```
[0, 100, -3, 3]
```



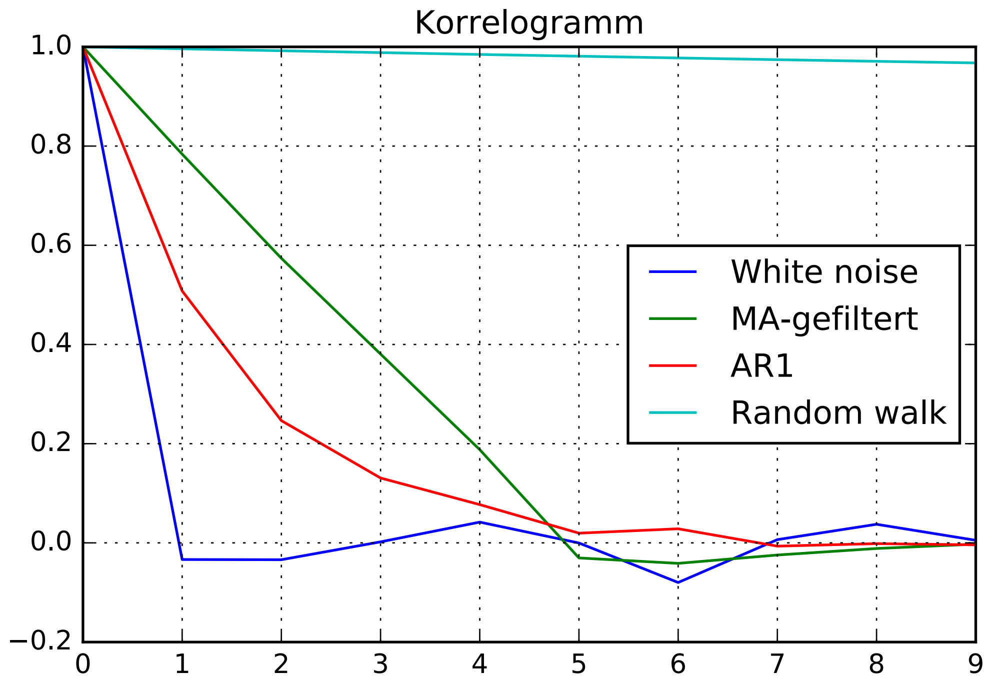
3.3 Autokorrelation

```
In [26]: def korrelation(x,y):
             mx=mean(x)
             my=mean(y)
             return sum( (x-mx)*(y-my) )/sqrt(sum( (x-mx)**2)* sum((y-my)**2) )

         print(korrelation(Y_white_noise,Y_L_W3))
         print(korrelation(Y_white_noise,Y_white_noise))
         print(korrelation(Y_L,Y_AR1))
```

```
0.347522360468
1.0
-0.019946912798
```

```
In [6]: def autokorrelation(x,l):
            r=ones(l)
            for i in range(1,l):
                x1=x[i:]
                x2=x[:-i]
                r[i]=korrelation(x1,x2)
            return r
        L=10
        plot(autokorrelation(Y_white_noise,L),label='White noise')
        plot(autokorrelation(Y_L_W5,L),label='MA-gefiltert')
        plot(autokorrelation(Y_AR1,L),label='AR1')
        plot(autokorrelation(Y_random_walk,L),label='Random walk')
        legend(loc='center right')
        grid()
        title('Korrelogramm')
```

Out[6]: <matplotlib.text.Text at 0x7f47ad9b3630>



3.4 Vorsicht bei Autokorrelationen!

Weisen die Daten Autokorrelationen auf, sind sie nicht mehr unabhängig! Die Anzahl der Freiheitsgrade reduziert sich. Dies muss bei der Hypothesentests berücksichtigt werden.

Siehe Matlab-Skript corr2.m von Ian Eisenmann http://eisenman.ucsd.edu/

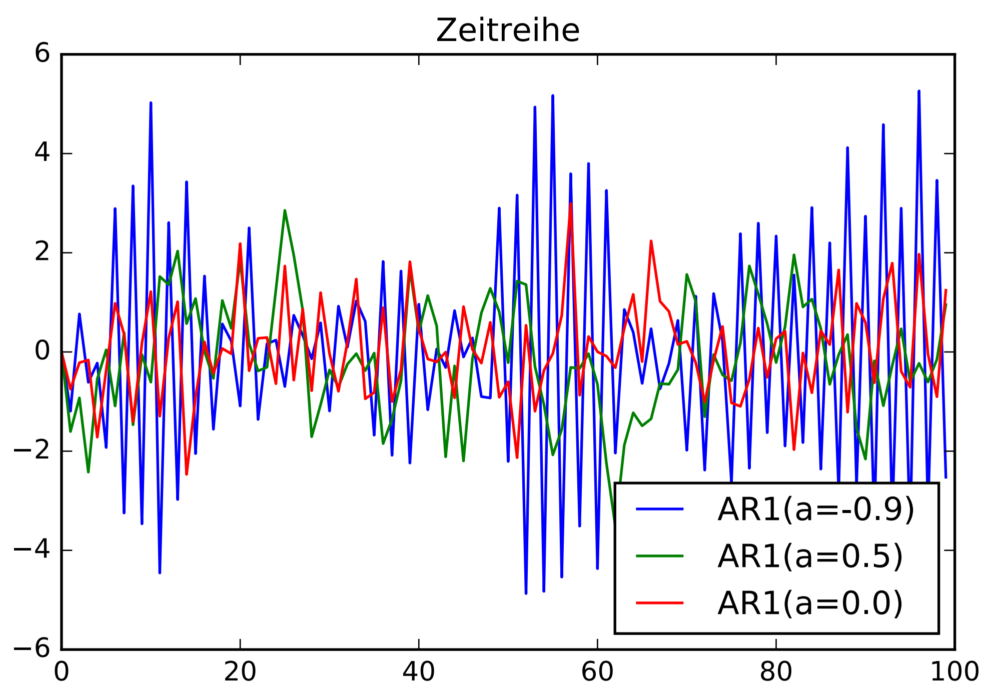
```
% Linear correlation (RHO) between X and Y in which the p-value (PVAL)
% accounts for autocorrelation in both records by using AR(1) processes as
% the null hypothesis rather than white noise as in the standard corr()
% function. Uses common method for estimating effective degrees of freedom
% in autocorrelated data (Bartlett, J. Roy. Stat. Soc. 98, 536-543, 1935;
% Mitchell et al., Climatic Change, WMO Technical Note 79, 1966; Bretherton
% et al., J. Climate 12, 1990-2009, 1999).
```

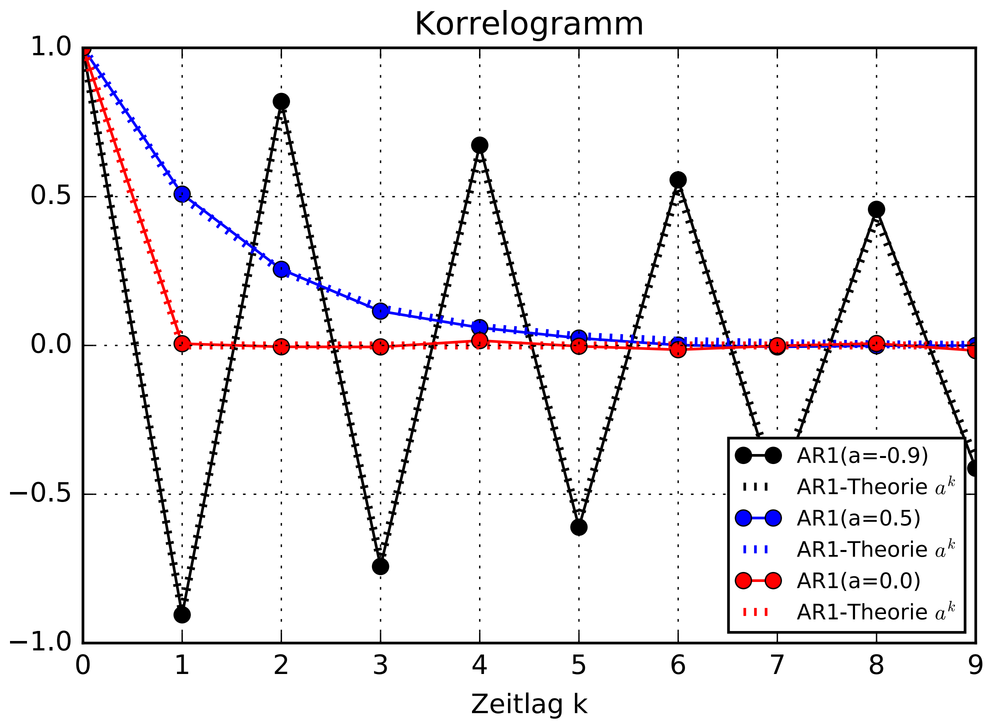
4 Beispiel AR1-Spektrum

In [27]:
```
%pylab inline
%config InlineBackend.figure_format = 'svg'


close('all')

figure(1)
figure(2)
N=10000
L=10
```

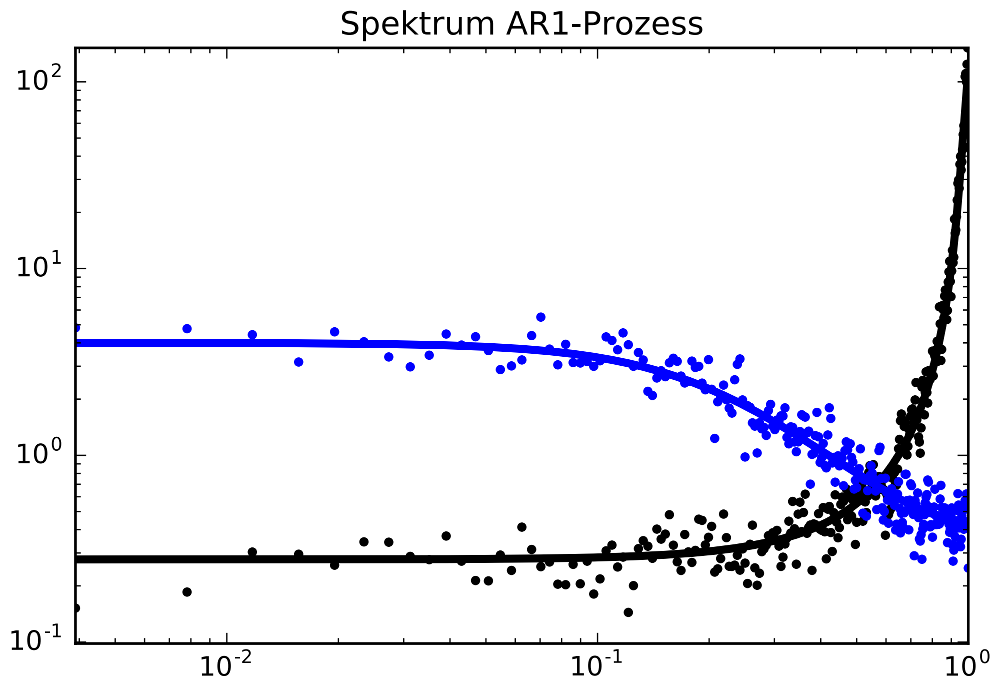
```
k=arange(0,L)
b=1.0
an=[-0.9,0.5,0.0]
Na=len(an)
Ya=zeros((Na,N))

c='kbrcy'
for i,a in enumerate(an):
    Y=AR1(N,a,b)
    Ya[i]=Y
    figure(1)
    plot(Y[0:100],label='AR1(a='+str(a)+') ')
    figure(2)
    plot(k,autokorrelation(Y,L),'o-',label='AR1(a='+str(a)+') ',color=c[i])
    plot(k,a**k,':',lw=3,label='AR1-Theorie $a^k$',color=c[i])
figure(1)
legend(loc='lower right')
title('Zeitreihe')
figure(2)
legend(loc='lower right',fontsize=8)
title('Korrelogramm')
xlabel('Zeitlag k')
grid()
```

```
Populating the interactive namespace from numpy and matplotlib
```






```
In [29]: def AR1_Spektrum(a,F,f):
             G=F/(1+a**2-2*a*cos(f*pi))
             return G

         figure()
         Pxx1,freqs=psd(Ya[0],NFFT=512)
         Pxx2,freqs=psd(Ya[1],NFFT=512)

         close()

         G1=AR1_Spektrum(an[0],b**2,freqs)
         loglog(freqs,G1,'-',linewidth=3,color=c[0])
         loglog(freqs,Pxx1,'.',color=c[0])

         G2=AR1_Spektrum(an[1],b**2,freqs)
         loglog(freqs,G2,'-',linewidth=3,color=c[1])
         loglog(freqs,Pxx2,'.',color=c[1])


         title('Spektrum AR1-Prozess')
         axis('tight')

Out[29]: (0.00390625, 1.0, 0.098249035829564429, 151.96640043431412)
```



Spektrum AR1-Prozess
10^2
10^1
10^0
10^-1
10^-2
10^-1
10^0